AUTOMATED SPACECRAFT DOCKING
USING A VISION-BASED RELATIVE NAVIGATION SENSOR

A Thesis

by

JEFFERY CHRISTOPHER MORRIS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2009

Major Subject: Aerospace Engineering

AUTOMATED SPACECRAFT DOCKING
USING A VISION-BASED RELATIVE NAVIGATION SENSOR

A Thesis

by

JEFFERY CHRISTOPHER MORRIS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	John Valasek
Committee Members,	John L. Junkins
	Srinivas R. Vadali
	Reza Langari
Head of Department,	Dimitris Lagoudas

August 2009

Major Subject: Aerospace Engineering

ABSTRACT

Automated Spacecraft Docking Using a
Vision-Based Relative Navigation Sensor. (August 2009)
Jeffery Christopher Morris, B.S., The University of Alabama
Chair of Advisory Committee: Dr. John Valasek

Automated spacecraft docking is a concept of operations with several important potential applications. One application that has received a great deal of attention recently is that of an automated docking capable unmanned re-supply spacecraft. In addition to being useful for re-supplying orbiting space stations, automated shuttles would also greatly facilitate the manned exploration of nearby space objects, including the Moon, near-Earth asteroids, or Mars. These vehicles would allow for longer duration human missions than otherwise possible and could even accelerate human colonization of other worlds. This thesis develops an optimal docking controller for an automated docking capable spacecraft. An innovative vision-based relative navigation system called VisNav is used to provide real-time relative position and orientation estimates, while a Kalman post-filter generates relative velocity and angular rate estimates from the VisNav output. The controller's performance robustness is evaluated in a closed-loop automated spacecraft docking simulation of a scenario in circular lunar orbit. The simulation uses realistic dynamical models of the two vehicles, both based on the European Automated Transfer Vehicle. A high-fidelity model of the VisNav sensor adds realism to the simulated relative navigation measurements. The docking controller's performance is evaluated in the presence of measurement noise, with the cases of sensor noise only, vehicle mass errors plus sensor noise, errors in vehicle moments of inertia plus sensor noise, initial starting position errors plus sen-

sor noise, and initial relative attitude errors plus sensor noise each being considered. It was found that for the chosen cases and docking scenario, the final controller was robust to both types of mass property modeling errors, as well as both types of initial condition modeling errors, even in the presence of sensor noise. The VisNav system was found to perform satisfactorily in all test cases, with excellent estimate error convergence characteristics for the scenario considered. These results demonstrate preliminary feasibility of the presented docking system, including VisNav, for space-based automated docking applications.

To my wife, Sarah Beth. It's finally finished!

ACKNOWLEDGMENTS

There are several people deserving of special mention for their contributions to this thesis. First, I would like to thank my advisor and thesis committee chair Dr. John Valasek. We have made it through so many difficulties together to reach this point, and his guidance has often proven invaluable along the way. I am also grateful to my committee members, whose technical expertise improved the quality of this research. Specifically, thanks go to Dr. John Junkins for his assistance in formulating the chosen spacecraft docking scenario; to Dr. Srinivas Vadali, whose suggestions and advice helped me develop both the reaction wheel models and successful controller design used in this work; and to Dr. Reza Langari for helping me understand the complexities of controller design theory. My gratitude goes as well to those members of "Team VisNav" who have gone before me, whose contributions include VisNav documentation, model code, graphics files, and other useful resources. I am deeply grateful to Matt and Cortney Ferguson, whose friendship and support in many forms throughout the years have been invaluable. I also appreciate all of the other friends and loved ones who have been there for me as well – emotionally, financially, and otherwise – during this journey. But most of all, I thank my wife. Without her love, patience, support, and encouragement, I simply never would have finished.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

FIGURE Page

FIGURE Page

FIGURE Page

FIGURE Page

FIGURE Page

FIGURE Page

FIGURE Page

FIGURE Page

CHAPTER I

INTRODUCTION

Automated spacecraft docking is an important capability for many reasons. It can introduce a new paradigm for construction of large space vehicles or structures by allowing modular designs; the sections could be launched separately and then assembled in Earth orbit via a series of docking maneuvers. This would greatly reduce the launch costs associated with transporting large structures into orbit. Automated docking also enables unmanned re-supply missions to manned outposts, such as the International Space Station (ISS) or other orbiting structures. Fleets of unmanned re-supply vehicles would greatly facilitate the manned exploration of nearby space objects, including the Moon, near-Earth asteroids, or Mars; these shuttles would need automated docking capabilities for many likely re-supply scenarios. Many other previously inaccessible new concepts of operation become possible by using automated on-orbit spacecraft docking.

The docking operational concept for spacecraft has existed since the early years of the Space Age. The Russians and Americans each independently performed successful on-orbit docking maneuvers between space vehicles in the 1960's [4]. The Russians employed a standardized, largely automated system that could operate with humans only having a supervisory role, but it also had a complete set of pilot controls to allow human intervention if necessary [5]. The Americans, on the other hand, opted for a series of 'one-off' docking schemes that were unique to each mission, rather than using a standard system design. American docking operations also required a human to be in the control loop at all times and the systems used a low level of automation [5].

This thesis follows the style of *IEEE Transactions on Automatic Control*.

This trend has continued to present day in American spacecraft docking operations, as a human-controlled spacecraft docking is performed each time a NASA space shuttle visits the ISS.

The Russians first demonstrated automated docking, which is performing a docking maneuver without a human in the control loop at all, by performing an automated docking of unmanned Cosmos vehicles 186 and 188 in October 1967. The vehicles successfully docked and remained in that configuration for three and a half hours, then separated and successfully executed their respective re-entry commands [6]. The Russians continue to have success in conducting on-orbit automated spacecraft dockings, even to present day; they docked various vehicles with the Mir space station while it orbited, and the ISS is also regularly visited by Russian Progress vehicles that dock automatically [7].

The Japanese became the second space program to successfully perform an on-orbit automated dock in 1998. They demonstrated completely autonomous rendezvous and docking operations between two spacecraft during space mission ETS-VII; the only means of human involvement in the exercises was a ground link for monitoring purposes [8]. While the Japanese have not yet attempted to repeat this on-orbit automated docking success, they are re-using some of the technology it demonstrated in the development of an automated shuttle called the H-II Transfer Vehicle (HTV) that is currently scheduled to launch in late 2009 [9]. This vehicle will regularly fly unmanned re-supply missions to the ISS. If the project succeeds, it will be a significant accomplishment for many reasons; but it will be even more commendable due to the relative lack of Japanese space-faring heritage as compared to other nations.

Since the turn of the century, attempts to master the automated spacecraft docking capability have greatly increased in number. American space agency NASA unsuccessfully attempted a proximity operations technology demonstration mission

in 2005 called DART (Demonstration of Autonomous Rendezvous Technology) [10], but partnered with the Defense Advanced Research Projects Agency (DARPA) for a successful and more comprehensive rendezvous and docking demonstration named Orbital Express in 2007 [11]. The European Space Agency launched the first of their own unmanned ISS re-supply shuttles, named the Automated Transfer Vehicle (ATV), in March 2008 [12]. The ATV docked to the ISS on April 3, 2008 using vision-based relative navigation sensors, after employing Differential GPS to get within close range [13]. Due to extensive redundancy and fault tolerance, the ATV is largely autonomous as well as completely automated, having no pilot controls; the only means of human intervention are remote emergency interrupt and abort capabilities [14].

The following research develops and tests a controller for an automated spacecraft docking scenario while using a vision-based system to provide relative navigation estimates. This is achieved by developing a computer simulation of a docking scenario in orbit around the Moon, with the chaser having a nominal starting position of 50 meters behind the target vehicle. The simulation uses realistic dynamical models of the two vehicles to be docked, both of which are based on the European Automated Transfer Vehicle (ATV). The numerical simulation also incorporates a high-fidelity model of VisNav, an innovative vision-based relative navigation sensor, to generate relative position and orientation six degrees-of-freedom (6-DOF) estimates. The designed docking controller is implemented in the simulation, and its performance in the presence of measurement noise and specific types of modeling errors is evaluated. It was found that the controller is capable of achieving automated docking while relying upon the realistic simulated relative navigation estimates provided by VisNav, even when other sources of modeling error are present. This verifies expected results from the control theory, while also demonstrating the feasibility of using the VisNav sensor system for spacecraft docking operations.

CHAPTER II

SPACECRAFT DOCKING

On-orbit spacecraft docking is a critical capability for a modern-day space program. Without it, the concept of an inhabited orbiting space station would not exist, since it would be impossible to re-supply the station or relieve the crew. Without docking capabilities, the Moon landings as we know them would not have happened since the lander would not have been able to reconnect to the orbiting module and return safely to Earth. These are just two examples that illustrate how different the space program would be if we did not possess on-orbit spacecraft docking capabilities.

This chapter discusses the spacecraft docking problem. Section A defines the spacecraft docking maneuver in the general context of spacecraft relative navigation. Governing equations for the particular set of coordinate frames selected for this work are also presented here. Section B describes different sensor packages that may be used by a spacecraft requiring relative navigation information. Finally, Section C presents issues and considerations specific to automated spacecraft docking.

A. Relative Navigation and Spacecraft Docking

Generally, relative navigation of spacecraft involves knowing one vehicle's position and orientation with respect to another (target) vehicle. Using an estimate of the current relative navigation solution, the chaser vehicle is then directed from some initial position and orientation relative to the target vehicle to some desired final relative position and orientation with respect to the target. For spacecraft docking relative navigation maneuvers, the target and chaser vehicles are both moving. In typical docking scenarios, the two vehicles are either in the same or very similar orbits, with the target vehicle ahead of the chaser in the orbit.

Polites defines spacecraft docking as mechanically connecting two vehicles by propelling a chase vehicle into a target vehicle with a nonzero linear velocity [7]. In other words, the docking maneuver involves directing the chaser to a zero final position relative to the target [15]. One can think of spacecraft docking as a controlled, low-speed collision that causes a mating device to trigger and mechanically connect the two vehicles upon contact. The final velocity of the chaser must be suitably slow in order to ensure orderly contact that does not cause damage to either vehicle. In fact, the entire docking maneuver is generally done at a very slow pace, so that it can be more easily aborted if a problem arises.

The governing equations for expressing relative vehicle translation are typically derived using Hill coordinates for the vehicle body frames, where x is positive radially outward from the orbit center, positive y denotes motion along the vehicle orbit track, and the z-coordinate describes motion out of the orbit plane in accordance with the right-hand rule. Using this frame convention for the equation derivation yields a well-known result called the Clohessy-Wiltshire equations [15]. However, coordinate conventions other than the traditional Hill frame have often been chosen for space-based operations as convenience or operational necessity dictated. For example, the ISS Body Coordinate System, which is internationally recognized, follows a Local-Vertical, Local-Horizontal (LVLH) based convention of the x-axis positive fore, y-axis positive starboard, and z-axis positive radially towards Earth [16]—the same body frame convention used for this project. This convention was adopted for this research primarily because of the relative navigation sensor model being used; the model has a proven track record with this body frame convention from its use in autonomous aerial refueling research as in Refs. [17] and [2].

Following the spirit of the derivations in [15], but assuming only that the target vehicle is non-maneuvering in a perfectly circular orbit while the chaser vehicle is able

to apply both control forces and control torques, the translational relative navigation equations in the target body frame are as follows:

$$\ddot{x} + \left(\frac{\mu}{{R_c}^3} - n^2\right) x - 2n\dot{z} = \frac{T_x}{m} \tag{2.1a}$$

$$\ddot{y} + \left(\frac{\mu}{{R_c}^3}\right) y = \frac{T_y}{m} \tag{2.1b}$$

$$\ddot{z} + \left(\frac{\mu}{{R_c}^3} - n^2\right) z + 2n\dot{x} + \mu\left(\frac{{R_c}^3 - R^3}{R^2 {R_c}^3}\right) = \frac{T_z}{m} \tag{2.1c}$$

where μ is the gravitational constant of the planetary body being orbited, n is the mean orbital rate of the target vehicle, m is the mass of the chaser, R is the target orbit radius, R_c is the current chase orbit radius, and (T_x, T_y, T_z) are the target frame components of the chaser vehicle's thrust vector.

Using the LVLH frame convention and 3-2-1 Euler angles to express the relative orientation of the chaser with respect to the target, one can also determine the governing equations for relative orientation between two orbiting vehicles. These equations are simply Euler's rotational equations of motion as expressed in principal body axes [18]:

$$\dot{\psi} = \frac{1}{\cos\theta}\left[w_y \sin\phi + w_z \cos\phi\right] \tag{2.2a}$$

$$\dot{\theta} = \frac{1}{\cos\theta}\left[w_y \cos\theta\cos\phi - w_z \cos\theta \sin\phi\right] \tag{2.2b}$$

$$\dot{\phi} = \frac{1}{\cos\theta}\left[w_x \cos\theta + w_y \sin\theta\sin\phi + w_z \sin\theta\cos\phi\right] \tag{2.2c}$$

$$\dot{w_x} = \left(\frac{I_{yy} - I_{zz}}{I_{xx}}\right) w_y w_z + \frac{L_x}{I_{xx}} \tag{2.3a}$$

$$\dot{w_y} = \left(\frac{I_{zz} - I_{xx}}{I_{yy}}\right) w_x w_z + \frac{L_y}{I_{yy}} \tag{2.3b}$$

$$\dot{w_z} = \left(\frac{I_{xx} - I_{yy}}{I_{zz}}\right) w_x w_y + \frac{L_z}{I_{zz}} \tag{2.3c}$$

where (ψ, θ, ϕ) are the 3-2-1 Euler orientation angles (yaw, pitch, and roll) of the

chaser relative to the target, (w_x, w_y, w_z) are the body frame relative angular velocities, (I_{xx}, I_{yy}, I_{zz}) are the principal inertias of the chaser vehicle in the body frame, and (L_x, L_y, L_z) are the body frame components of the chaser's applied torque vector.

B. Relative Navigation Sensors

Automated spacecraft docking requires successfully employing several important capabilities simultaneously; however, one technical issue truly critical to the maneuver's success or failure is the quality of the relative navigation (rel-nav) information used to direct it. High solution rate and highly accurate rel-nav information, even in the face of mitigating environmental conditions or poor scenario geometry, are both important for successful docking. Achieving all of these things simultaneously requires a very sophisticated rel-nav sensor.

The relative navigation (consisting of both relative position and relative orientation) information between two spacecraft must be accurately measured in real-time, and successfully relayed to the command computer, in order to accomplish safe maneuvering of the vehicles relative to each other. This requires a sensor package onboard the vehicles that can estimate the instantaneous 6-DOF relative navigation quantities at a useful rate and accuracy. The most common system used to accomplish this in near-earth situations is the Global Positioning System (GPS). Basically, this system uses a constellation of several satellites to triangulate the current position of a receiver unit. The accuracy and solution rate using GPS varies widely depending on the application and especially the processing done on the received satellite signals, but real-world on-orbit errors are generally on the order of meters [19]. Thus, relative GPS (RGPS), currently the most accurate type of GPS, is considered to be an acceptable rel-nav solution for general automated relative spacecraft maneuvers in Low

Earth Orbit (LEO). However, RGPS is not accurate enough by itself to be used for docking, and in any case the GPS signals only reach out to about Geosynchronous Earth Orbit (GEO) [19]. Therefore, GPS is not a viable relative navigation option in deep space or in orbit around other objects such as the Moon.

Another class of relative navigation sensors rely primarily on pattern or image recognition capabilities to provide the relative 6-DOF estimate. This type of sensor often pairs a camera with software that extracts and tracks features from successive images, by which an estimate of the motion of the vehicle relative to its target can be derived. Reference [20] discusses a specific example of this type of sensor package; it describes how researchers from CalTech and the Jet Propulsion Laboratory have experimented with using an image-based motion estimation (IBME) algorithm in conjunction with an inertial measurement unit (IMU) for relative navigation. The researchers fed the outputs from both the IBME algorithm and the IMU into an Indirect Kalman Filter to generate relative 6-DOF estimates. This technique was found to improve both estimate accuracy and robustness when compared to basic image recognition. Estimate update rate was implied to be no more than 4 Hz using this technique, but estimate errors were as low as on the order of centimeters or millimeters using an experimental testbed. As this experimental sensor matures, it could become a viable relative navigation sensor solution for general automated docking operations, both in Earth orbit and beyond.

A third major technique for estimating rel-nav information is to use laser range finders. As the name implies, lasers are shone from one vehicle to another, which reflects them back to a receiver on the first vehicle. A relative position estimate can then be determined at long range, with full 6-DOF possible at shorter ranges. One sensor that implements this operational concept is the Rendezvous and Docking Sensor (RVS) from Jena-Optronik GmbH in Germany [21]. This sensor is flight-

proven, having flown on Space Shuttle Missions STS-84 and STS-86, which docked with the Mir Space Station in 1997, and on the maiden flight of the unmanned European ATV ISS re-supply vehicle in 2008. It also will be used as the primary rel-nav sensor for the unmanned Japanese HTV which is currently scheduled to fly its first mission in late 2009. According to the company's data sheet for the sensor, it is capable of accuracies of one centimeter at short range and of less than half a meter at long range. During the planning for the ATV-1 mission, the European Space Agency reported that they expected the sensor to accurately estimate range and direction (3-DOF) to a true range of 1000 meters, while generating full 6-DOF estimates at ranges of 40 meters or less [13]. The RVS sensor package certainly appears to have demonstrated its viability as an acceptable relative navigation solution for automated docking operations.

Yet another kind of rel-nav system relies on the real-time detection of customized, active light beacons to estimate the relative 6-DOF state. One sensor in this class is VisNav, which was developed in the late 1990's by researchers at Texas A&M University [22]. VisNav has already been established as a viable rel-nav sensor technology in general by researchers at Texas A&M, who have primarily focused on using the sensor for aerial refueling applications [17, 23, 24, 25, 26]. One aim of this current research is to demonstrate the feasibility of using VisNav for space-based automated docking maneuvers as well. The next chapter will describe the VisNav system, its algorithms, and some potential applications of the system in detail.

C. Automated Spacecraft Docking

Though many docking systems over the years have required a human in the control loop, some systems have successfully docked without any human assistance or control.

This capability becomes necessary when having a human on-board is infeasible and/or communications lag is too great for remote-controlled operation [27]. However, an on-orbit automated docking is difficult to successfully perform, as evidenced by the fact that only the Russians have had continuous and repeated success in doing so [7].

Primarily what makes automated docking so difficult is the sheer complexity of replacing human senses, training, common sense, and decision-making ability with computerized systems. This must be accomplished while at the same time making the entire docking system robust and fault-tolerant enough to succeed on its own in a real-time scenario. Several space programs, in the United States and abroad, are currently trying to solve this major issue. The other requirements needed to accomplish automated docking include the actual docking mechanism and the relative navigation sensor. The docking mechanism for manned operations is an existing technology that could also be employed for automated docking. Several relative navigation sensors exist today that are probably 'good enough' to use for automated docking as well. Thus, solving the control system aspect of the problem clears what is probably the most formidable hurdle to having a successful automated docking system.

CHAPTER III

THE VISNAV SYSTEM

This chapter describes the VisNav (***Vis***ion-based ***Nav***igation) sensor system. Section A provides an overview of the system, some underlying theory, and a few potential applications for the sensor. VisNav generates its relative navigation estimates through the use of a non-linear least squares algorithm called Gaussian Least-Squares Differential Correction (GLSDC), which is presented in Section B. Finally, Section C presents a high-fidelity Matlab^{TM} model of the VisNav system which was used in the numerical simulations that support this work.

A. System Description

VisNav is a unique kind of relative navigation sensor. It is low-cost and low-power with low weight and volume requirements, yet produces near real-time relative 6-DOF (position and orientation) estimates with high precision and at a high data rate of 100 Hz [28]. The system operates using structured, active infrared beacons, analogous to radar, allowing it to perform satisfactorily even in high-disturbance ambient environments. The system has been established as a viable relative navigation sensor technology for automated aerial refueling applications by researchers at Texas A&M University [17, 23, 26], and it is expected to perform capably for spacecraft docking as well.

1. Overview

The VisNav system consists of four hardware modules as follows: a beacon constellation consisting of several (generally between eight and 16) frequency-modulated light-emitting diode (LED) beacons, a beacon controller module that orchestrates

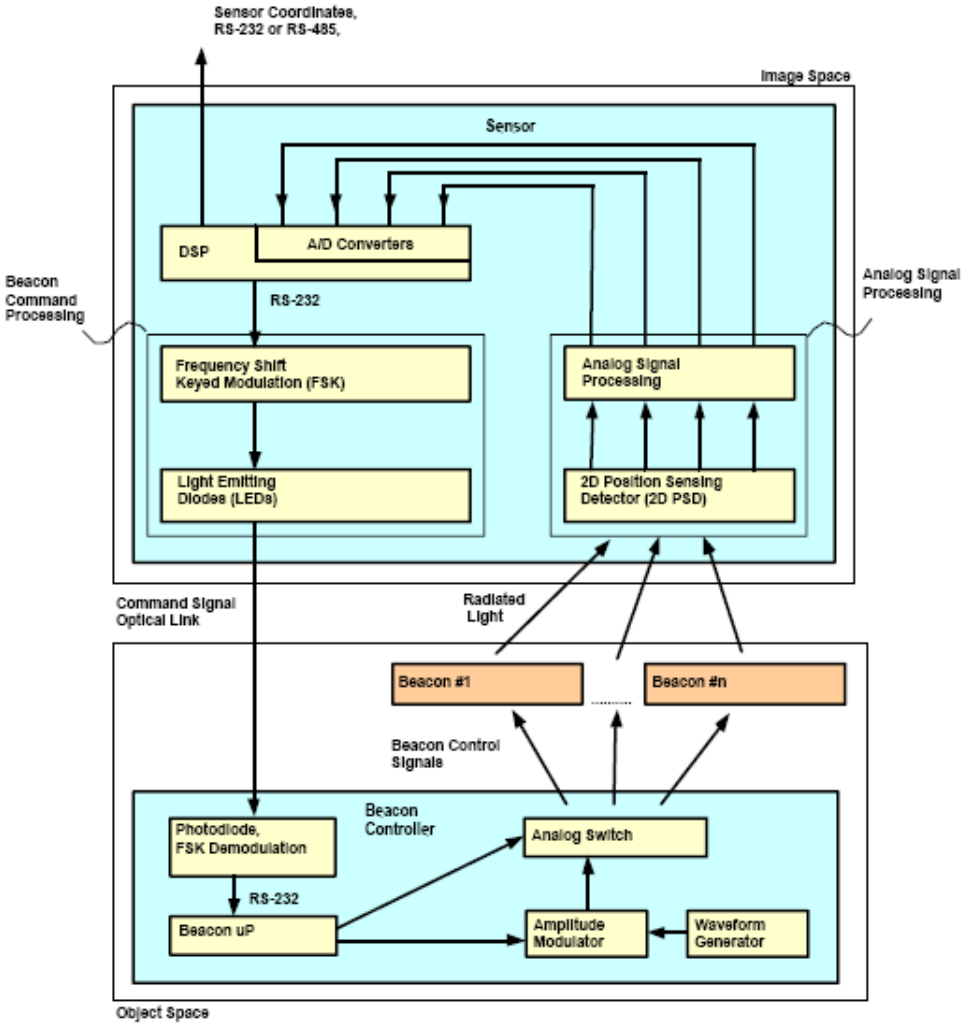

Fig. 1. VisNav System Architecture

the beacons, a sensor unit built around position-sensing diode (PSD) technology, and a central processing unit (CPU) to perform calculations. For spacecraft docking, the sensor and CPU are both mounted on the chaser vehicle, while the beacons and beacon controller are fixed onto the target vehicle. A low-power wireless data link allows communication between the CPU on the chaser and the beacon controller on the target. Figure 1 illustrates the interaction of the VisNav system modules.

VisNav generates an estimate of the relative position and orientation of the

sensor's image plane with respect to the target vehicle [29]; a constant coordinate transformation can then be employed to obtain the real-time relative 6-DOF information for the chaser vehicle with respect to the target. A brief description of the VisNav system is provided here; for more detailed information, the interested reader should consult Refs. [28], [29], and [30].

VisNav beacons are given specific wave-forms and are modulated in the frequency domain, analogous to radar. This allows for ease in discrimination of the beacons from ambient signals, even in a high-radiation environment. The beacons are mounted in fixed locations on the target vehicle, with known target frame coordinates. When a beacon is activated, some of its radiation falls onto the surface of the sensor's PSD by passing through a wide-angle lens. This causes four currents to flow, one from each of the four edges of the PSD surface, as shown in Figure 2. The beacon radiation centroid falling nearer to one edge than another causes an imbalance in the output currents; in fact, the difference in the current strengths from two opposite edges is almost linearly proportional to the distance of the light centroid from each of those edges [28]. Thus, VisNav can determine the location of the beacon radiation centroid on the sensor face based on the PSD currents generated. This location is then used to form a line-of-sight unit vector pointing from the sensor towards the beacon that originated the received energy.

The unit line-of-sight vectors corresponding to different beacons are collected and updated with each system cycle. The current relative position and orientation of the sensor with respect to the target vehicle can be robustly estimated when at least four unique beacons are concurrently in the sensor's field of view. VisNav uses the collinearity equations, which will be discussed next, and the GLSDC algorithm, presented in Section B, to generate least-squares relative 6-DOF estimates based on the current set of beacon line-of-sight unit vectors.

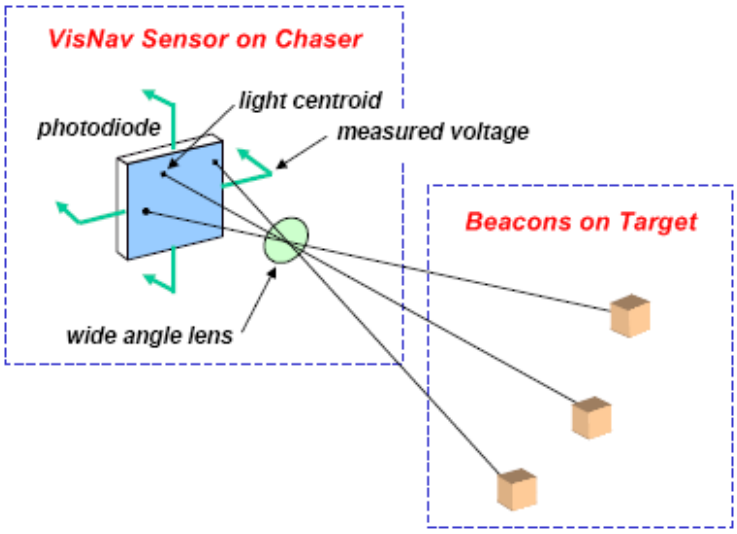

Fig. 2. Illustration of VisNav Operation

2. Measurement Model

VisNav's CPU generates relative 6-DOF estimates by assuming that the beacon measurements it receives are coming from an ideal pin-hole camera. For a pin-hole camera, the observed object, the pin-hole center, and the object image are collinear; thus the governing equations for such a measurement model are called the 'collinearity equations'. However, the VisNav sensor module is not a perfect sensor; there are both lens and PSD distortions in its beacon measurements. Therefore, a one-time, *a priori* calibration map of the sensor is generated to neutralize these distortion effects over the sensor's useful field of vision, so that the ideal pin-hole model can be applied to its measurements with validity [30].

Since VisNav's measurement model consists of the collinearity equations, they can be used to derive the relationship between the relative position of the sensor with respect to the target vehicle, and the corresponding beacon light centroid locations on the sensor face. This derivation follows the steps (and to an extent, the notation)

used in Ref. [31], though an abbreviated form of the same derivation can be found in any of the VisNav-related articles listed in the References section.

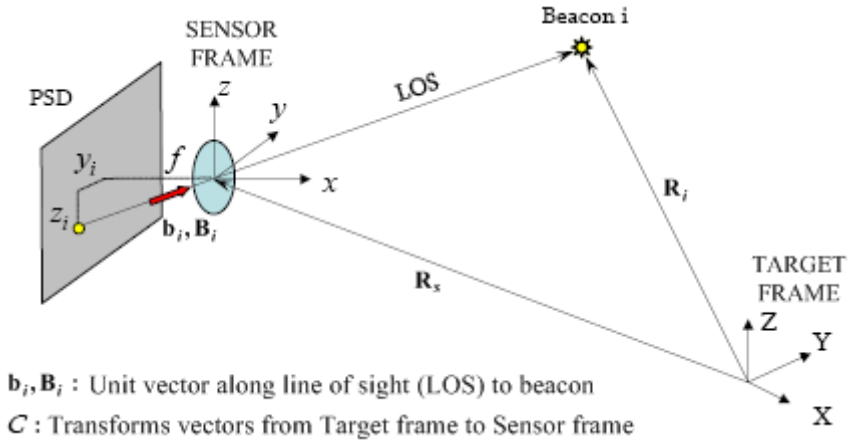

Fig. 3. VisNav Sensor Measurement Model [2])

Consider Figure 3, which illustrates the vector relationships between the target, a VisNav beacon, and the VisNav sensor. Two coordinate frames are shown in the figure, one fixed to the target vehicle and the other fixed to the center of the VisNav sensor's wide-angle lens. An unknown Direction Cosine Matrix (DCM), denoted as $\boldsymbol{C}$, transforms between the two frames. The i^{th} system beacon is shown, as are its position vectors with respect to the target and sensor frame origins. The sensor's position vector with respect to the target frame origin is also labelled, but is an unknown quantity that must be estimated by VisNav. The beacon is fixed to the target vehicle, so its position with respect to the target frame is known; however, since the sensor's position with respect to the target is unknown, the beacon's position with respect to the sensor is unknown as well.

The relative position of the sensor with respect to the target coordinate frame is defined as $\mathbf{R}_s$ as shown in the figure. In the target frame, the components of this

vector can be expressed as

$$\mathbf{R}_s = \begin{bmatrix} X_s \\ Y_s \\ Z_s \end{bmatrix} \tag{3.1}$$

The known location of the i^{th} beacon in target frame coordinates can be represented with the vector

$$\mathbf{R}_i = \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} \tag{3.2}$$

Referring back to Figure 3, notice that the x-axis of the sensor frame points along the boresight of the VisNav sensor; since the focal length f of the wide-angle lens is constant, the x-coordinate of any unit vector expressed in the sensor frame is therefore always f. This can be immediately applied to vector $\mathbf{b}_i$, which points along the line-of-sight vector from the sensor to the i^{th} beacon. Normalizing $\mathbf{b}_i$ into a unit vector and expressing it in sensor frame coordinates results in:

$$\mathbf{b}_i = \frac{1}{\sqrt{f^2 + y_i^2 + z_i^2}} \begin{bmatrix} f \\ -y_i \\ -z_i \end{bmatrix} \tag{3.3}$$

This same unit line-of-sight vector in target frame coordinates can be expressed as

$$\mathbf{B}_i = \frac{1}{\sqrt{(X_i - X_s)^2 + (Y_i - Y_s)^2 + (Z_i - Z_s)^2}} \begin{bmatrix} (X_i - X_s) \\ (Y_i - Y_s) \\ (Z_i - Z_s) \end{bmatrix} \tag{3.4}$$

if one recalls that the vector from the sensor to beacon i is equivalent to the difference of vectors $\mathbf{R}_i$ and $\mathbf{R}_s$.

The two forms of the unit line-of-sight vector shown in Equations (3.3) and (3.4)

are related via the unknown direction cosine matrix $\boldsymbol{C}$. While there are many options for parameterizing a DCM, some of which are discussed at length in Chapter 3 of Ref. [32], VisNav's solution logic uses Modified Rodrigues Parameters (MRPs) in its internal calculations. The MRP vector $\mathbf{p}$ is a three-member set of attitude parameters, and as with any three-element attitude parameter set, it has a singularity. However, MRPs are quite useful due to the fact that the singularity is at $\Phi = \pm 360\,^{\circ}$, so that any attitude motion *except* a complete revolution back to the starting point can be described without encountering the singularity [32]. The (3×1) MRP vector can be defined in terms of the principal rotation vector $\mathbf{e}$ and the principal rotation angle Φ as

$$\mathbf{p} = \tan\frac{\Phi}{4}\,\mathbf{e} \tag{3.5}$$

The direction cosine matrix $\boldsymbol{C}$ in terms of the MRPs is

$$\boldsymbol{C} = \boldsymbol{I} + \frac{8\left[\mathbf{p}\times\right]^2 - 4\left(1 - \mathbf{p}^T\mathbf{p}\right)\left[\mathbf{p}\times\right]}{\left(1 + \mathbf{p}^T\mathbf{p}\right)^2} \tag{3.6}$$

where $\boldsymbol{I}$ represents the (3×3) identity matrix, and the matrix $[\mathbf{p}\times]$ is skew-symmetric and defined as

$$\left[\mathbf{p}\times\right] = \begin{bmatrix} 0 & -p_3 & p_2 \\ p_3 & 0 & -p_1 \\ -p_2 & p_1 & 0 \end{bmatrix} \tag{3.7}$$

Referring back to Equations (3.3) and (3.4), which are related via the direction cosine matrix $\boldsymbol{C}$ as parameterized in (3.6), one can conclude that

$$\mathbf{b}_i = \boldsymbol{C} \cdot \mathbf{B}_i \tag{3.8}$$

Equation (3.8) is the unit vector form of the collinearity equations, which can be

solved for the beacon centroid location in sensor frame coordinates [29]:

$$y_i = -f\frac{C_{21}\left(X_i - X_s\right) + C_{22}\left(Y_i - Y_s\right) + C_{23}\left(Z_i - Z_s\right)}{C_{11}\left(X_i - X_s\right) + C_{12}\left(Y_i - Y_s\right) + C_{13}\left(Z_i - Z_s\right)} \tag{3.9a}$$

$$z_i = -f\frac{C_{31}\left(X_i - X_s\right) + C_{32}\left(Y_i - Y_s\right) + C_{33}\left(Z_i - Z_s\right)}{C_{11}\left(X_i - X_s\right) + C_{12}\left(Y_i - Y_s\right) + C_{13}\left(Z_i - Z_s\right)} \tag{3.9b}$$

$$i = 1, 2, \ldots N$$

where (y_i, z_i) are the image plane coordinates of the i^{th} beacon centroid, f is the constant focal length of the wide-angle lens, N is the total number of beacons in use, C_{ij} is the ij^{th} element of the unknown relative Direction Cosine Matrix $\boldsymbol{C}$, and other quantities are as previously defined. Thus, Equations (3.9) show how the sensor's relative position with respect to the target directly affects the resulting beacon light centroids detected by the sensor.

3. Applications

The VisNav relative navigation system holds considerable promise for space-based operations between cooperative vehicles for several reasons. First, the VisNav system is of small size and weight and is readily adaptable to a variety of vehicle systems. Also, VisNav has the ability to operate in a variety of lighting conditions, due to both active and passive radiation filtering and its frequency-modulated infrared beacons [29]. This is a clear advantage while in orbit, since lighting conditions constantly change as a vehicle's position with respect to the sun changes. Finally, the system has an expected operational range of more than 100 meters in the space environment. This means that VisNav should be capable of providing relative 6-DOF information for the entire terminal phase of a docking maneuver according to Polites's definition of the rendezvous and docking phases [7]. Once this range is demonstrated, it could allow a reduction in vehicle weight by elimination or consolidation of relative navigation

sensor packages. Such weight reduction would be a valuable side-effect of using VisNav for space-based operations.

VisNav also has several potential applications in a variety of other fields. Besides being useful in autonomous aerial refueling operations as previously mentioned, the sensor could be used as part of a remote piloting input system. Ref. [33] describes how one can use a glove embedded with miniature LED beacons, coupled with a pair of VisNav sensors mounted in a stereo configuration, to control a flight simulator. The operator tilts, rotates, and otherwise gestures his or her hand, which VisNav detects by sensing the motion of the beacons. These motions are then translated into corresponding commands such as throttle, yaw, or pitch to direct the simulated aircraft. It is not much of a stretch to see how this same setup could be applied to Unmanned Aerial Vehicles (UAVs) or other remotely piloted vehicle systems.

Another application that has been suggested for VisNav, but not yet investigated, is for use of the system in autolanding UAV helicopters onto an aircraft carrier deck. The beacons would be mounted onto the deck at known, fixed points, while the sensor would be carried on the vehicle and interface with the UAV's control system for operation. Many other applications for VisNav exist and are constantly being found by researchers. It is clear that this system has myriad possible uses, making it a truly valuable sensor technology.

B. GLSDC Algorithm

The Gaussian Least-Squares Differential Correction algorithm is well-suited for use with the VisNav sensor system due to several factors. It is a sequential estimator so it can be run in near real-time as the system requires. Also, since it is a type of least-squares algorithm, it does not tax VisNav's relatively modest on-board CPU

with its low computational requirements. Another useful feature of GLSDC is that it outputs an error covariance matrix along with its estimate, providing a measure of how well conditioned the given information was and thus how much the estimate can be trusted. This is extremely useful when running the VisNav sensor in tandem with a Kalman filter estimator as was done for this work, since the covariance matrix can be fed directly in to the Kalman filter along with the estimates to which it corresponds. This provides real-time guidance to the filter as to how accurate it should assume the VisNav estimates are. Chapter IV contains further discussion of how VisNav and the Kalman filter were integrated for this research.

GLSDC is intended to be used in situations in which a set of system measurements have been taken, a valid measurement model exists, and there are some unknown system parameters that must be estimated. The procedure used by the algorithm actually results in optimal parameter estimates, in a minimum error variance sense, for a given level of assumed measurement noise [34]. This discussion of GLSDC is derived primarily from material in Refs. [31] and [34], and should be considered an introduction to the algorithm. The interested reader is encouraged to consult [34] for a more in-depth treatment of GLSDC.

First, define the GLSDC state vector as the unknown quantities to be estimated by the algorithm. For VisNav, these are the relative position and orientation of the sensor with respect to the target, expressed in target frame coordinates. Recalling from the previous section that VisNav internally represents orientation with Modified Rodrigues Parameters, or MRPs, the state vector can be written as

$$\mathbf{x} = \begin{bmatrix} \mathbf{R}_s \\ \\ \mathbf{p} \end{bmatrix} \tag{3.10}$$

using the notation from Section A. Next, the set of N accumulated measurements,

which for VisNav consists of four or more unit line-of-sight vectors to currently visible beacons, is represented by the vector $\tilde{\mathbf{b}}$:

$$\tilde{\mathbf{b}} = \begin{bmatrix} \tilde{\mathbf{b}}_1 \\ \tilde{\mathbf{b}}_2 \\ \vdots \\ \tilde{\mathbf{b}}_N \end{bmatrix} \tag{3.11}$$

The measurement model associated with the measurements $\tilde{\mathbf{b}}$ is represented as $\mathbf{h}(\mathbf{x})$ in GLSDC notation; however, for VisNav, recall that the measurement model was derived in the previous section to be

$$\mathbf{b}_i = \boldsymbol{C} \cdot \mathbf{B}_i \tag{3.8}$$

Thus, it can be written that

$$\mathbf{b}_i = \mathbf{h}_i(\mathbf{x}) = \boldsymbol{C} \cdot \mathbf{B}_i \tag{3.12}$$

However, since these measurements were made by a real (non-ideal) sensor, measurement noise must be accounted for in the model. This results in

$$\tilde{\mathbf{b}}_i = \mathbf{h}_i(\mathbf{x}) + \mathbf{v}_i \tag{3.13}$$

for the measurement model equation, where the $(\tilde{\ })$ denotes $\tilde{\mathbf{b}}_i$ as a measured quantity.

The measurement noise $\mathbf{v}_i$ in Equation (3.13) is assumed to be zero-mean and Gaussian[1], with a covariance of $\mathbf{R}_i = \boldsymbol{E}\{\mathbf{v}_i \mathbf{v}_i^T\}$, where $\boldsymbol{E}$ is the expectation operator. According to Ref. [35], the expected value of a function $f(x)$ of a discrete random variable x is defined as $\boldsymbol{E}\{f(x)\} = \sum_j f(x(j))\, p(x(j))$, where $p(x(j))$ is the proba-

[1]The Gaussian or normal probability density function (pdf) is often expressed in mathematical notation as $p(x) \sim N(\mu, R)$, where $p(x)$ is the probability of x, μ is the distribution's mean, and R is the distribution's covariance.

bility of occurrence for variable $x(j)$. Thus, the measurement noise covariance matrix for the current set of N beacon measurements $\tilde{\mathbf{b}}_i$ is

$$\boldsymbol{R} = \begin{bmatrix} R_1 & 0 & 0 & 0 \\ 0 & R_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & R_N \end{bmatrix} \tag{3.14}$$

If the true relative position and orientation at any point in time is $\mathbf{x}$, then the estimated relative position and orientation is represented by $\hat{\mathbf{x}}$. While both the true and estimated relative 6-DOF variables can be translated into beacon line-of-sight vectors corresponding to their values via the measurement model from (3.12), the true beacon line-of-sight vectors are not perfectly known. Instead, they are known only by the measurements $\tilde{\mathbf{b}}_i$. Assuming the estimate is imperfect, there will be some error when the measured beacon line-of-sight vectors are compared to the ones derived from the estimate $\hat{\mathbf{x}}$. This difference is the residual estimate error $\Delta\boldsymbol{b}$, which is defined mathematically for N given measurements as

$$\Delta\mathbf{b} = \begin{bmatrix} \tilde{\mathbf{b}}_1 - \mathbf{h}_1(\hat{\mathbf{x}}) \\ \tilde{\mathbf{b}}_2 - \mathbf{h}_2(\hat{\mathbf{x}}) \\ \vdots \\ \tilde{\mathbf{b}}_N - \mathbf{h}_N(\hat{\mathbf{x}}) \end{bmatrix} \tag{3.15}$$

The goal of the GLSDC algorithm is to minimize the residual error $\Delta\boldsymbol{b}$. This goal is equivalent to minimizing a cost function defined as the weighted sum of the squares of the residuals:

$$\boldsymbol{J} = \frac{1}{2}\Delta\mathbf{b}^T\boldsymbol{W}\Delta\mathbf{b} \tag{3.16}$$

where $\boldsymbol{W}$ is a diagonal, positive definite matrix of residual weighting coefficients.

However, the optimal estimate that minimizes the cost function $\boldsymbol{J}$ cannot be explicitly determined for this case due to the nonlinear nature of the VisNav measurement model. Instead, the minimization must be performed in an iterative, least-squares fashion that employs local linearization techniques.

The optimal estimate can be approximated as a current guess plus a differential correction:

$$\hat{\mathbf{x}} = \mathbf{x}_c + \Delta\mathbf{x} \tag{3.17}$$

If $\Delta\mathbf{x}$ is appropriately small, one can linearize the measurement model around the current guess $\mathbf{x}_c$ using a first-order Taylor series approximation:

$$\mathbf{h}\left(\hat{\mathbf{x}}\right) \approx \mathbf{h}\left(\mathbf{x}_c\right) + \boldsymbol{H}\Delta\mathbf{x} \tag{3.18}$$

Note that this approximation requires the matrix $\boldsymbol{H} \in \Re^{3N\times 6}$, which contains the derivatives of the measurement model equations with respect to the GLSDC states, evaluated at the current guess. This is known as the measurement sensitivity matrix, with rows defined as

$$\boldsymbol{H} = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_N \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathbf{h}_1}{\partial \mathbf{x}} \\ \frac{\partial \mathbf{h}_2}{\partial \mathbf{x}} \\ \vdots \\ \frac{\partial \mathbf{h}_N}{\partial \mathbf{x}} \end{bmatrix} \tag{3.19}$$

where the partial derivatives with respect to $\mathbf{x}$ can be partitioned into separate derivatives with respect to position and orientation:

$$\frac{\partial \mathbf{h}_i}{\partial \mathbf{x}} = \left[\frac{\partial \mathbf{h}_i}{\partial \mathbf{R}_s} : \frac{\partial \mathbf{h}_i}{\partial \mathbf{p}} \right] \tag{3.20}$$

These separated partial derivatives can then be expressed as:

$$\frac{\partial \mathbf{h}_i}{\partial \mathbf{R}_s} = -\boldsymbol{C}\left\{\boldsymbol{I}_{3\times 3} - \mathbf{B}_i\mathbf{B}_i^T\right\}/\sqrt{\left(X_i - X_s\right)^2 + \left(Y_i - Y_s\right)^2 + \left(Z_i - Z_s\right)^2} \tag{3.21a}$$

$$\frac{\partial \mathbf{h}_i}{\partial \mathbf{p}} = \frac{4}{\left(1 + \mathbf{p}^T\mathbf{p}\right)^2}\left[\mathbf{b}_i\times\right]\left\{\left(1 - \mathbf{p}^T\mathbf{p}\right)\boldsymbol{I}_{3\times 3} - 2\left[\mathbf{p}\times\right] + 2\mathbf{p}\mathbf{p}^T\right\} \tag{3.21b}$$

Now that the measurement model has been locally linearized, the expressions for the residual error and the cost function $\boldsymbol{J}$ must be approximated as well. For the current guess $\mathbf{x}_c$, the approximate residual is simply

$$\Delta\mathbf{b}_c \equiv \tilde{\mathbf{b}} - \mathbf{h}\left(\mathbf{x}_c\right) \tag{3.22}$$

The full residual, calculated after the current guess is updated using the differential correction, becomes

$$\Delta\mathbf{b} \approx \tilde{\mathbf{b}} - \mathbf{h}\left(\mathbf{x}_c\right) - \boldsymbol{H}\Delta\mathbf{x} = \Delta\mathbf{b}_c - \boldsymbol{H}\Delta\mathbf{x} \tag{3.23}$$

The cost function from (3.16) is rewritten in terms of the linearized full residual errors as

$$\boldsymbol{J}_P = \frac{1}{2}\left(\Delta\mathbf{b}_c - \boldsymbol{H}\Delta\mathbf{x}\right)^T\boldsymbol{W}\left(\Delta\mathbf{b}_c - \boldsymbol{H}\Delta\mathbf{x}\right) \tag{3.24}$$

where the weighting matrix $\boldsymbol{W}$ is now assigned as the reciprocal of the measurement error covariance matrix, or $\boldsymbol{W} = \boldsymbol{R}^{-1}$, as prescribed by minimum variance weighted least squares [34]. Equation (3.24) is the cost function that GLSDC actually minimizes through its iterative procedure.

Finding the minimum of the approximate cost function from (3.24) involves taking the partial derivative of the function with respect to the differential correction $\Delta\mathbf{x}$:

$$\frac{\partial \boldsymbol{J}_p}{\partial\left(\Delta\mathbf{x}\right)} = \boldsymbol{H}^T\boldsymbol{W}\boldsymbol{H}\Delta\mathbf{x} - \boldsymbol{H}^T\boldsymbol{W}\Delta\mathbf{b}_c \tag{3.25}$$

Setting (3.25) equal to zero and solving for $\Delta\mathbf{x}$ yields

$$\Delta\mathbf{x} = \left(\boldsymbol{H}^T\boldsymbol{W}\boldsymbol{H}\right)^{-1}\boldsymbol{H}^T\boldsymbol{W}\Delta\mathbf{b}_c \tag{3.26}$$

which is the optimal differential correction that minimizes the linearized residual errors. This is the magnitude of the correction that should be applied to the current guess $\mathbf{x}_c$ during the current GLSDC iteration.

The GLSDC iterative solution procedure is implemented for VisNav as follows. GLSDC is initialized for the first timestep with a generic starting guess for the relative position and orientation. GLSDC iteratively updates this guess based on the measurements it is given and the optimal differential corrections it calculates, until a specified stopping criterion is met (such as a desired estimate convergence or a maximum number of iterations). The converged estimate is then returned to VisNav as the relative 6-DOF estimate for the first timestep. At the second timestep, GLSDC receives an updated set of measurements from VisNav; it uses the converged estimate from the first timestep as its initial guess, then iterates on the new measurements until a stopping criterion is reached. The second timestep's converged estimate is returned to VisNav as the new current relative 6-DOF estimate, and the procedure repeats until the final time is reached. Figure 4 contains a summary of the GLSDC algorithm.

As previously mentioned, an estimation error covariance matrix can be obtained from GLSDC in addition to its converged relative navigation estimate. This covariance matrix is related to how well conditioned the given measurement set was and can be considered a measure of the quality of the final 6-DOF estimate. The error covariance matrix comes straight from the steps of the GLSDC algorithm, as it is the quantity $\left(\boldsymbol{H}^T\boldsymbol{W}\boldsymbol{H}\right)^{-1}$ that forms part of the optimal differential correction calculated during each iteration. Thus, the operator can receive automatic updates

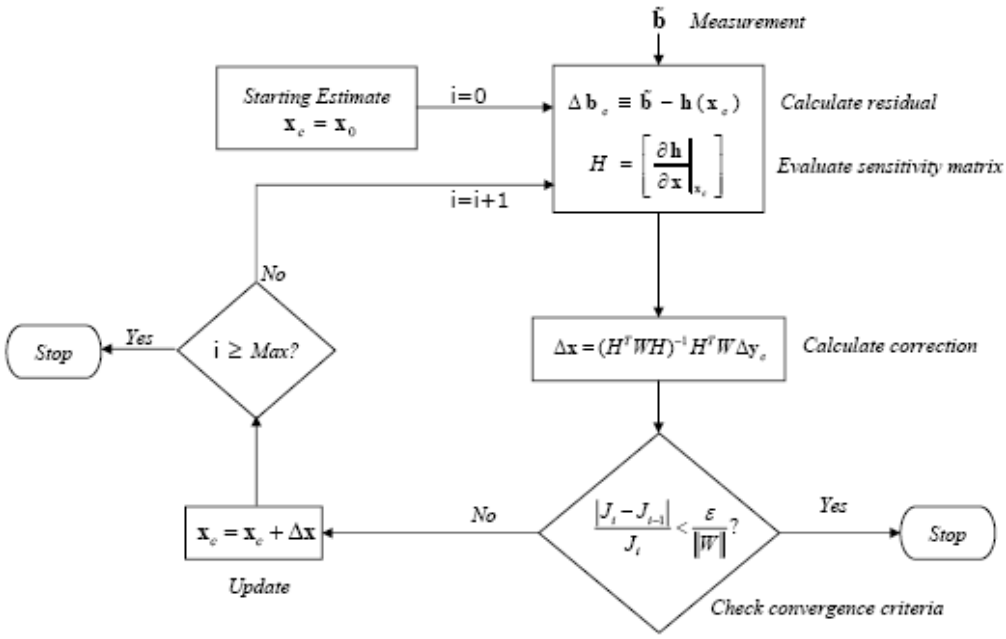

Fig. 4. Gaussian Least Squares Differential Correction Algorithm [2]

about the solution quality by extracting this matrix at each timestep along with the algorithm's converged estimate. This property of the GLSDC algorithm is certainly very useful, especially when using it in concert with a Kalman filter as was done in this work.

C. High-fidelity Sensor Model

A high-fidelity model of the VisNav sensor system has been developed in Matlab^{TM}; Figure 5 shows a functional block diagram of the sensor model. The model requires a user-specified beacon configuration suitable for the vehicle of interest upon initialization. Then, at every time-step, the model receives the true inertial position and orientation of the target and chaser vehicles being simulated. This information is used in real-time to accurately simulate the generation of individual beacon radiation

centroid locations on the sensor face. It does this by calculating the ideal locations of the centroids using the ideal pin-hole camera model discussed in Section A, and then applying in reverse the *a priori* calibration mapping also discussed in Section A. The model next adds noise with the same characteristics as in the real system to the resulting centroid locations. Unit line-of-sight vectors to each beacon are calculated from the simulated centroids, just as in the real system, and then fed to a GLSDC function to obtain real-time relative position and orientation estimates, also as in the real system. These simulated 6-DOF estimates are the primary outputs of the sensor model, and they are returned at a rate of 100 Hz in order to match the VisNav hardware's performance.

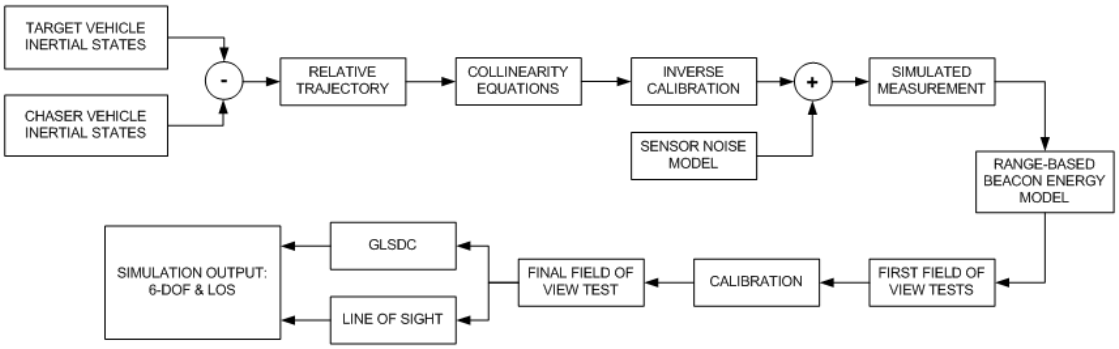

Fig. 5. VisNav Sensor Model Functional Block Diagram (Image courtesy of StarVision Technologies, Inc.)

It is important to note that the sensor model carefully segregates the truth (inertial) inputs from the simulated measurements generated from them (which include simulated sensor noise as mentioned previously), so that the solution modules are using only the simulated beacon measurements to obtain the 6-DOF estimates. This ensures that the model does not have any 'advantage' over the real sensor system, but that its estimates are obtained in the same way and using only the same information that is available to the real hardware.

Development work on this sensor model continues as needed in order to keep pace with advances in the physical VisNav system. Thus, the model used in this research is of somewhat greater complexity and fidelity to the real hardware than the version of this model that was used and validated in Refs. [2] and [17]. Since all model modifications were verified against the hardware as they were made, the VisNav sensor model used in this work should be accepted as a reliable emulator of the VisNav system.

CHAPTER IV

ESTIMATION FOR VISNAV SPACECRAFT APPLICATIONS

This chapter discusses the Kalman filter used in this work to generate sequential velocity and angular rate estimates based on the output of the VisNav sensor system discussed in Chapter III. Section A provides the development of the particular type of Kalman filter used in this research. Section B discusses how the resulting filter is applied to the general relative navigation problem. Finally, the tuning of the Kalman filter for use with the VisNav sensor for spacecraft rel-nav applications is discussed in Section C.

A. Discrete-Time Linear Kalman Filter

Estimators are often used to infer values for the states of a dynamic system based on a given system model, a set of known inputs, and a set of measured outputs over a specific time interval. When the estimates are generated in real-time, based on previous and current system measurements, the estimator is known as a sequential estimator. The Kalman filter is one type of state estimator that has been found to be useful in a wide variety of applications [1]. This development of a sequential discrete-time linear Kalman filter primarily draws from material in Refs. [1] and [2], and the notation used here is consistent with those references.

A discrete-time sequential estimator's primary purpose is to provide a current estimate of the system's states, $\hat{\mathbf{x}}_k$, where the 'hat' ($\hat{}$) denotes an estimated quantity, based on a current measurement of some combination of the system's states $\tilde{\mathbf{y}}_k$. To do this, an estimator uses a two-step process of prediction and correction [1]. First, the sequential estimator *predicts* the value of the states vector $\hat{\mathbf{x}}$ at time k by propagating a known system model from the previous timestep $(k-1)$, using the known applied

control vector $\mathbf{u}$ acting on the system. Then, when the k^{th} measurement is delivered to the estimator, it *corrects* its previous prediction for the states at step k based on how far the predicted states deviated from the actual measured states at that step.

Assume a linear system, represented mathematically as

$$\mathbf{x}_{k+1} = \mathbf{\Phi}_k \mathbf{x}_k + \mathbf{\Gamma}_k \mathbf{u}_k + \mathbf{\Upsilon}_k \mathbf{w}_k \tag{4.1}$$

where $\mathbf{x}_k \in \Re^n$ is the vector of system states, $\mathbf{u}_k \in \Re^m$ is the vector of system control inputs, $\mathbf{w}_k \in \Re^p$ is the process noise vector, and the subscript k denotes the k^{th} timestep. The quantities $\mathbf{\Phi}_k \in \Re^{n \times n}$, $\mathbf{\Gamma}_k \in \Re^{n \times m}$, and $\mathbf{\Upsilon}_k \in \Re^{n \times p}$ are called the state transition matrix, the control distribution matrix, and the disturbance matrix respectively. The process noise vector $\mathbf{w}_k$ can represent any unmodeled system dynamics, or some unknown disturbance force acting on the system, with no restrictions placed on its characteristics at this time. The system's measurement model equation in discrete time is

$$\tilde{\mathbf{y}}_k = \boldsymbol{H}_k \mathbf{x}_k + \mathbf{v}_k \tag{4.2}$$

where $\tilde{\mathbf{y}}_k \in \Re^r$ is some measured combination of the system states, $\mathbf{v}_k \in \Re^r$ is the measurement noise, $\boldsymbol{H}_k \in \Re^{r \times n}$ is the system output matrix, and subscript k again denotes the k^{th} timestep.

For this assumed system model, a generic estimator's coupled prediction and correction procedures can be expressed as

$$\hat{\mathbf{x}}_{k+1}^- = \mathbf{\Phi}_k \hat{\mathbf{x}}_k^+ + \mathbf{\Gamma}_k \mathbf{u}_k \tag{4.3}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \boldsymbol{K}_k \left[\tilde{\mathbf{y}}_k - \boldsymbol{H}_k \hat{\mathbf{x}}_k^- \right] \tag{4.4}$$

for a time-varying update gain $\boldsymbol{K}$. Since the '$-$' and '$+$' superscripts denote quantities before and after the correction respectively, it should be clear that Equation (4.3)

represents the prediction process, while the correction procedure is shown in Equation (4.4). These equations describe any discrete sequential estimator in general, though some estimators employ a constant update gain $\boldsymbol{K}$ instead of a time-varying one. The primary differentiator between types of estimators is the process used to select the update gain.

The Kalman filter selects an optimal time-varying update gain; the gain is optimal in the sense that it produces a minimum variance estimation error *at each time step* [2]. The Kalman filter selects $\boldsymbol{K}$ by assuming certain stochastic properties for both the measurement and process noise vectors, then it places the poles of the estimator based on these properties [1]. The process noise vector $\mathbf{w}_k$ is assumed to be a zero-mean, Gaussian white-noise process, which means that it is not correlated with itself forward or backward in time:

$$\boldsymbol{E}\{\mathbf{w}_k\mathbf{w}_j^T\} = \begin{cases} 0 & k \neq j \\ \boldsymbol{Q}_k & k = j \end{cases} \tag{4.5}$$

The measurement noise is also assumed to be a Gaussian white-noise process, and therefore not self-correlated in time:

$$\boldsymbol{E}\{\mathbf{v}_k\mathbf{v}_j^T\} = \begin{cases} 0 & k \neq j \\ \boldsymbol{R}_k & k = j \end{cases} \tag{4.6}$$

Notice that for MIMO (multi-input, multi-output) systems, both $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$ are matrix quantities, and are called the process (or modeling) error covariance matrix and the measurement error covariance matrix respectively. Finally, the process and measurement noise vectors are assumed to be uncorrelated with each other for all time, so that $\boldsymbol{E}\{\mathbf{v}_k\mathbf{w}_k\} = 0$ for all k. The equation for the Kalman filter update gain can now be derived based on these assumed noise characteristics; doing so will also

reveal the rest of the algorithm for the discrete-time linear Kalman filter.

The estimation error at time k is defined as the estimated state minus the true state and is denoted with a ($\tilde{}$) symbol:

$$\tilde{\mathbf{x}}_k \equiv \hat{\mathbf{x}}_k - \mathbf{x}_k \tag{4.7}$$

Next, define the estimation error covariance at time k as the expectation of the squared sum of the estimation errors:

$$\boldsymbol{P}_k \equiv \boldsymbol{E}\left\{\tilde{\mathbf{x}}_k \tilde{\mathbf{x}}_k^T\right\} \tag{4.8}$$

Note that this results in a matrix quantity such that $\boldsymbol{P}_k \in \Re^{n\times n}$ since $\mathbf{x}_k \in \Re^n$ as previously mentioned. A pair of equations for prediction and correction of this covariance matrix can be obtained. The elements required to produce them include the estimator state prediction and correction equations from (4.3) and (4.4), the linear system model defined in (4.1) and (4.2), and the previously assumed stochastic properties of $\mathbf{w}_k$ and $\mathbf{v}_k$. The resulting equations are analogous to the state prediction-correction equations used by the general discrete estimator, and are central to the Kalman filter's unique procedure for update gain selection. From Ref. [1], the covariance matrix prediction-correction equations are:

$$\boldsymbol{P}_{k+1}^- = \boldsymbol{\Phi}_k \boldsymbol{P}_k^+ \boldsymbol{\Phi}_k^T + \boldsymbol{\Upsilon}_k \boldsymbol{Q}_k \boldsymbol{\Upsilon}_k^T \tag{4.9}$$

for the covariance prediction (or propagation) step and

$$\boldsymbol{P}_k^+ = \left[\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right] \boldsymbol{P}_k^- \tag{4.10}$$

for the covariance correction step. Please see Ref. [1] for the details of how to obtain these equations.

Now that all of the required assumptions and definitions are in place, the ex-

pression for the Kalman filter update gain can be readily obtained. Recall that the Kalman filter selects the gain at each timestep to be optimal in a minimum estimation error variance sense. The cost function corresponding to this optimization is equivalent to the trace of the error covariance matrix after the correction step [1]:

$$\boldsymbol{J}\left(\boldsymbol{K}_k\right) = \operatorname{tr}\left(\boldsymbol{P}_k^+\right) \tag{4.11}$$

The update gain that minimizes this cost function can be found by taking the derivative of $\boldsymbol{J}$ with respect to $\boldsymbol{K}_k$, setting the expression equal to zero, and solving for $\boldsymbol{K}_k$. Employing some appropriate matrix trace identities from Chapter 2 of [36] while constraining both $\boldsymbol{P}_k^-$ and $\boldsymbol{R}_k$ to be symmetric matrices yields

$$\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{K}_k} = -2\left(\boldsymbol{I} - \boldsymbol{K}_k\boldsymbol{H}_k\right)\boldsymbol{P}_k^-\boldsymbol{H}_k^T + 2\boldsymbol{K}_k\boldsymbol{R}_k \tag{4.12}$$

as an expression for the derivative of $\boldsymbol{J}$ with respect to $\boldsymbol{K}_k$. Setting this equation equal to zero and solving for $\boldsymbol{K}_k$ results in the desired expression for the optimal Kalman update gain:

$$\boldsymbol{K}_k = \boldsymbol{P}_k^-\boldsymbol{H}_k^T\left[\boldsymbol{H}_k\boldsymbol{P}_k^-\boldsymbol{H}_k^T + \boldsymbol{R}_k\right]^{-1} \tag{4.13}$$

Recall that $\boldsymbol{R}_k$ is the measurement error covariance matrix. This matrix should be provided or calculated at each timestep as an indicator of the received measurement's accuracy. The final pieces of information needed to complete the Kalman filter algorithm are the initial values for both the state vector and estimation error covariance matrix, which are required to begin the sequential calculations. The state vector should be initialized with a starting guess in the near neighborhood of expected operation,

$$\hat{\mathbf{x}}\left(t_0\right) = \hat{\mathbf{x}}_0 \tag{4.14}$$

and the estimation error covariance should be assigned a value based upon the quality of the state vector starting guess, according to the relationship

$$\boldsymbol{P}_0 = \boldsymbol{E}\left\{\tilde{\mathbf{x}}\left(t_0\right)\tilde{\mathbf{x}}\left(t_0\right)^T\right\} \tag{4.15}$$

Provided the initial error covariance and starting guess are not severely mismatched, the Kalman filter will converge within just a few steps to a very good estimate for the current values of the states.

Therefore, given the initial conditions for both the states and the estimation error covariance matrix, the Kalman filter sequentially propagates the estimates until it receives the first measurement. Then, it calculates the optimal update gain for that timestep and corrects its predictions. Finally, it propagates the corrected state estimates and covariance matrix forward to the next timestep, where it receives the next measurement and the process is repeated. A summary of the discrete-time linear Kalman filter algorithm is presented in Table I.

B. Kalman Filter Design for Relative Navigation

The previous section began the derivation of the discrete-time linear Kalman filter by starting from the assumption that model and measurement equations describing a system of interest were available. This section will serve to develop the model and measurement equations for relative navigation that are required to apply the discrete-time linear Kalman filter to automated spacecraft docking.

There are at least two options for generating Kalman filter model equations for relative navigation. First, one could linearize the full relative navigation equations presented in Chapter II to obtain a linear dynamical model of the system. This choice captures the full dynamics of the spacecraft relative navigation problem but

Table I. Discrete-Time Linear Kalman Filter [1, 2]

Model	$\mathbf{x}_{k+1} = \boldsymbol{\Phi}_k \mathbf{x}_k + \boldsymbol{\Gamma}_k \mathbf{u}_k + \boldsymbol{\Upsilon}_k \mathbf{w}_k$ $\tilde{\mathbf{y}}_k = \boldsymbol{H}_k \mathbf{x}_k + \mathbf{v}_k$
Noise	$\mathbf{w}_k \sim \boldsymbol{N}(\mathbf{0}, \boldsymbol{Q}_k)$ $\mathbf{v}_k \sim \boldsymbol{N}(\mathbf{0}, \boldsymbol{R}_k)$
Initialize	$\hat{\mathbf{x}}(t_0) = \hat{\mathbf{x}}_0$ $\boldsymbol{P}_0 = \boldsymbol{E}\{\tilde{\mathbf{x}}(t_0)\,\tilde{\mathbf{x}}(t_0)^T\}$
Gain	$\boldsymbol{K}_k = \boldsymbol{P}_k^- \boldsymbol{H}_k^T \left[\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{R}_k\right]^{-1}$
Update	$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \boldsymbol{K}_k \left[\tilde{\mathbf{y}}_k - \boldsymbol{H}_k \hat{\mathbf{x}}_k^-\right]$ $\boldsymbol{P}_k^+ = \left[\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right] \boldsymbol{P}_k^-$
Propagation	$\hat{\mathbf{x}}_{k+1}^- = \boldsymbol{\Phi}_k \hat{\mathbf{x}}_k^+ + \boldsymbol{\Gamma}_k \mathbf{u}_k$ $\boldsymbol{P}_{k+1}^- = \boldsymbol{\Phi}_k \boldsymbol{P}_k^+ \boldsymbol{\Phi}_k^T + \boldsymbol{\Upsilon}_k \boldsymbol{Q}_k \boldsymbol{\Upsilon}_k^T$

uses relatively complex equations to do so. The resulting filter is also limited with respect to potential application beyond the current research, because its modeled dynamics are only valid for spacecraft relative navigation scenarios. Trying to apply the filter to a dynamically different type of relative navigation situation, such as aerial refueling of aircraft, would require completely different model equations.

Another way to develop the Kalman filter model equations is to focus only on the kinematics of relative navigation—that is, to enforce the relationships between the relative position and orientation states and their derivatives. This enables a much simpler model development process, while still yielding good Kalman filter performance. The resulting filter is more general and can be applied to *any* relative navigation scenario, since the kinematic relationships between states and their derivatives are

the same for any relative navigation application—even if the dynamics are different. In fact, a Kalman filter basically identical to the one presented here has already been successfully applied (in concert with VisNav) to autonomous aerial refueling in Ref. [2], even though the dynamics of aerial refueling are very different than for spacecraft docking. This demonstrates how a Kalman filter designed in this way only needs to be re-tuned for each new relative navigation application rather than being completely redesigned each time. The proper way to tune the Kalman filter for a given relative navigation scenario will be discussed in the next section.

For this research, the Kalman filter model equations were developed using the second (enforcing kinematics) design option. This enables the possibility that the resulting Kalman filter can be used again with the VisNav sensor in some future relative navigation simulation, without needing to conduct a major filter redesign. Also, the Kalman filter was used primarily as a velocity estimator in this work, with less emphasis placed on its ability to improve VisNav's relative state estimates. This made the kinematics modeling design option an even more logical choice. This model equation development closely follows the one described in Chapter IV of Ref. [2], which used a Kalman filter with VisNav in an autonomous aerial refueling scenario.

First, define the state vector for the Kalman filter model in continuous time as follows:

$$\mathbf{x}(t) \equiv \begin{bmatrix} \mathbf{z}(t) \\ \dot{\mathbf{z}}(t) \\ \ddot{\mathbf{z}}(t) \end{bmatrix} \tag{4.16}$$

where $\mathbf{z}(t)$ is a vector composed of the relative position and orientation states according to the relation

$$\mathbf{z}(t) \equiv \begin{bmatrix} \mathbf{r}_{rel}(t) \\ \theta(t) \end{bmatrix} \tag{4.17}$$

Consistent with conventions used elsewhere in this work, orientation is expressed here in 3-2-1 Euler angles, which are collectively denoted as $\theta(t)$. Thus, the Kalman filter states vector is $\mathbf{x}(t) \in \Re^{18\times 1}$, consisting of the relative position and orientation states, the relative velocity and angular velocity states, and the relative acceleration and angular acceleration states, respectively.

Next, one assumption about the kinematics of the relative navigation scenario of interest will be made. The relative accelerations and relative angular accelerations between the two vehicles will all be assumed constant, so that their derivatives $\dddot{\mathbf{r}}_{rel}(t)$ and $\dddot{\theta}(t)$, collectively denoted as $\dddot{\mathbf{z}}(t)$, are zero. This assumption necessarily introduces some error into the kinematical model, which will be captured by the use of the model process noise variable $\mathbf{w}(t)$ introduced in the previous section. Since apart from this assumption, the modeled kinematical relationships between the relative states and their derivatives are exact, the model equations for the Kalman filter can be expressed in continuous time as

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{\mathbf{z}}(t) \\ \ddot{\mathbf{z}}(t) \\ \dddot{\mathbf{z}}(t) \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \boldsymbol{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{z}(t) \\ \dot{\mathbf{z}}(t) \\ \ddot{\mathbf{z}}(t) \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \boldsymbol{I} \end{bmatrix} \mathbf{w}(t) \tag{4.18}$$

where $\boldsymbol{I}$ is the (6×6) identity matrix and $\mathbf{0}$ is a (6×6) matrix of zeros. Notice that a control input $\mathbf{u}(t)$ term is not present since the model is of the kinematics only. Again, Equation (4.18) exactly represents the kinematic relationships between the position, velocity, and acceleration states, so the only modeling errors are a result of the constant accelerations assumption.

In order for Equation (4.18) to be implemented in the Kalman filter, it must first be converted into discrete time. Doing so yields:

$$\mathbf{x}_{k+1} = \boldsymbol{\Phi}_k \mathbf{x}_k + \boldsymbol{\Upsilon}_k \mathbf{w}_k \tag{4.19}$$

where the matrices $\boldsymbol{\Phi}_k$ and $\boldsymbol{\Upsilon}_k$ are equal to

$$\boldsymbol{\Phi}_k = \begin{bmatrix} \boldsymbol{I} & (t_{k+1} - t_k) \cdot \boldsymbol{I} & \frac{1}{2}(t_{k+1} - t_k)^2 \cdot \boldsymbol{I} \\ \mathbf{0} & \boldsymbol{I} & (t_{k+1} - t_k) \cdot \boldsymbol{I} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{I} \end{bmatrix} \tag{4.20}$$

and

$$\boldsymbol{\Upsilon}_k = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \boldsymbol{I} \end{bmatrix} \tag{4.21}$$

respectively. Notice that (4.19) matches the form of (4.1) as it should, accounting for a zero-coefficient control input term. The formulation of the discrete-time Kalman filter requires that measurements be received at uniform intervals; this means that $(t_{k+1} - t_k)$ is constant. In fact, since VisNav delivers measurements once every 0.01 seconds, or at 100 Hz, then $(t_{k+1} - t_k) = 0.01$ for all timesteps k. This means that Equation (4.19) is a linear, time-invariant discrete model, greatly simplifying the propagation calculations required of the Kalman filter.

Now, it is necessary to determine the measurement model equation for the Kalman filter. VisNav, the relative navigation sensor used in this research, generates an estimate of its own relative position and relative orientation with respect to the target vehicle. Assuming VisNav measures imperfectly, as does every real sensor, then the VisNav measurements can be represented in discrete time as

$$\tilde{\mathbf{y}}_k = \boldsymbol{H}_k \mathbf{x}_k + \mathbf{v}_k = \begin{bmatrix} \boldsymbol{I} & \mathbf{0} & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{z}(t) \\ \dot{\mathbf{z}}(t) \\ \ddot{\mathbf{z}}(t) \end{bmatrix} + \mathbf{v}_k \tag{4.22}$$

in the measurement model equation standard form introduced in the previous section. Recall that the Kalman filter also requires an estimate of the magnitude of the mea-

surement error in the form of a measurement error covariance matrix $\boldsymbol{R}_k \equiv \boldsymbol{E}\{\mathbf{v}_k \mathbf{v}_k^T\}$. VisNav produces this matrix implicitly because it uses the Gaussian Least-Squares Differential Correction solution algorithm to calculate its estimates as described in Chapter III.

The final step in the Kalman filter design process is tuning its estimate convergence to behave properly for a specific application. The estimate convergence properties are primarily determined by the relative weighting of the measurement error (via the matrix $\boldsymbol{R}_k$) and the modeling error (via the matrix $\boldsymbol{Q}_k$). However, $\boldsymbol{R}_k$ should not be altered since VisNav automatically provides the values of this matrix that correspond to its estimates. So, the filter's convergence is adjusted by tuning the values in matrix $\boldsymbol{Q}_k$ to achieve the desired estimate behavior. Tuning the $\boldsymbol{Q}_k$ matrix is discussed in the next section.

C. Kalman Filter Tuning for Automated Spacecraft Docking Using VisNav

Tuning the Kalman filter formulated in Section B is necessarily application dependent, since it involves adjusting the values in the (constant) model error covariance matrix $\boldsymbol{Q}_k$. Recall from Section A that this matrix is symmetric and positive definite; it is also diagonal, since it is defined as

$$\boldsymbol{Q}_k = \boldsymbol{E}\{\mathbf{w}_k \mathbf{w}_k^T\} \tag{4.23}$$

Altering this matrix tells the Kalman filter how valid the constant acceleration assumption is that was used in Equation (4.19). Since this assumption certainly does not apply to the same degree for all possible relative navigation scenarios, the need for tuning matrix $\boldsymbol{Q}_k$ becomes clear.

The most expeditious way to tune $\boldsymbol{Q}_k$ is via simulation of the chosen relative nav-

igation maneuver. Using simulation is valuable because both the estimated Kalman filter states and the true states are known, unlike in a real-world scenario in which only the estimates would be known perfectly. In simulation, the estimates and the true values can be compared; their difference can then be used to facilitate the tuning process. This is generally done by calculating the estimation error, defined in (4.7), plotting its time history, and superimposing the corresponding so-called 3-σ error bounds on the plots for comparison. The 3-σ bounds corresponding to the estimate error of a Kalman filter can be calculated by multiplying the square root of the diagonal elements of matrix $\boldsymbol{P}_k$ by a factor of three:

$$3\text{-}\sigma \text{ bounds } = \; 3 \cdot \sqrt{\text{diag}\left(\boldsymbol{P}_k\right)} \tag{4.24}$$

The 3-σ error bounds represent three standard deviations from the mean of a probability distribution function. For a Gaussian process, the probability that the estimate error lies within the 3-σ error bounds at any moment in time is 99.7% [35]. While in this case the modeling error $\mathbf{w}_k$ is actually not Gaussian, the 3-σ bounds are still indicative of the filter's performance. Therefore, the 3-σ bounds remain a useful guide for the tuning process. Once the filter is tuned, there will be good correlation between the 3-σ error bounds and the Kalman filter's estimate error. Then, the 3-σ bounds can be used as a measure of the filter's performance even in real-world applications, where it is not possible to actually know the true estimate error.

The procedure for tuning $\boldsymbol{Q}_k$ is as follows, according to Ref. [2]. The translational and rotational terms are assigned separate weighting parameters, due to the differing orders of magnitude in the relative translational and rotational accelerations:

$$\boldsymbol{Q}_k = \begin{bmatrix} q_{pos} \cdot \boldsymbol{I} & \mathbf{0} \\ \mathbf{0} & q_{rot} \cdot \boldsymbol{I} \end{bmatrix} \tag{4.25}$$

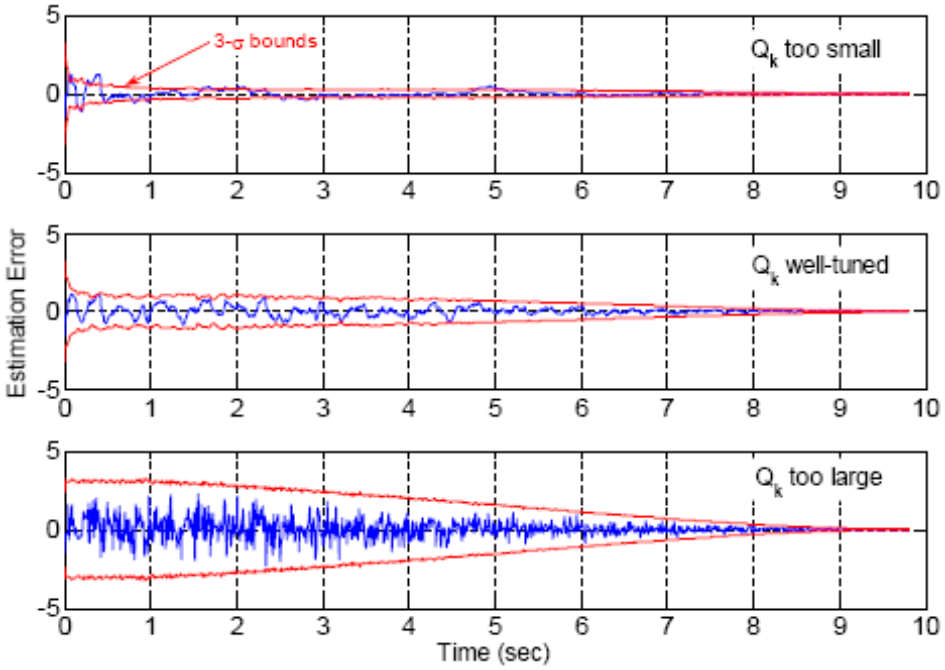

Fig. 6. Example of the Kalman Filter Parameter Tuning Procedure [2]

where $\boldsymbol{I}$ is the (3×3) identity matrix and $\mathbf{0}$ is a (3×3) matrix of zeros. Next, the Kalman filter's estimate error time histories are plotted along with their corresponding 3-σ error bounds. The values of q_{pos} and q_{rot} are then adjusted until the error bounds fit well around the estimate error plots. Then, once the designer is satisfied with the results, the filter is ready for use in the relative navigation application of interest. This tuning procedure is illustrated in Figure 6. Notice how in the first graph, the error bounds are too small and the estimate error exceeds them in several places. In the third graph, the error bounds are too large, showing that the Kalman filter is being too conservative in generating its estimates. The middle graph exhibits a well-tuned $\boldsymbol{Q}_k$ matrix.

For the current research, the Kalman filter designed in Section B was tuned in this same way. Automated spacecraft docking runs were performed in simulation, and the Kalman filter's estimate error results were plotted with their 3-σ error bounds for

each run. The constant parameters q_{pos} and q_{rot} were then iteratively adjusted until the error bounds fit the estimate errors as well as possible. The final values of the parameters were then utilized to design and test the docking controller developed in the next chapter. The final tuning parameter values were chosen to be 75 and 160 for q_{pos} and q_{rot}, respectively, because these values gave the best correlation between the 3-σ bounds and the estimate errors for this application.

CHAPTER V

SPACECRAFT DOCKING CONTROLLER

Successful controller design for automated docking applications requires consideration of several issues. First, a design technique must be selected that can adequately compensate for likely disturbances and sensor noise. Environmental effects, off-nominal mass properties, and other modeling uncertainties are all likely sources of disturbance. The controller must also be designed around the capabilities and limitations of the vehicle being used for the docking maneuver, so that its performance demands on the vehicle are achievable. Most importantly, the candidate controller should be a design that is implementable using measurements actually available to the vehicle, or else the design will not be feasible for the given scenario. Finally, any candidate controller design should be thoroughly tested via simulation before being implemented in any other form, in order to minimize risks to man or machine.

This chapter presents the docking controller designed for this project. Section A defines the control objectives for the design procedure. A variation of the Sampled-Data Linear Quadratic Regulator (SDLQR) control scheme was developed for this research; Section B contains the basic SDLQR derivation. Finally, Section C describes each of the two ways the SDLQR control theory was implemented for this project.

A. Control Objective

The automated docking scenario used in this experiment assumes that rendezvous has already taken place. The target and chaser vehicles are therefore in the same orbit, which is circular, with the target leading the chaser in the orbit by a constant distance. Since the orbit is circular, the vehicles both have the same velocity magnitude prior to the docking maneuver and are effectively at rest relative to each

other. The vehicles also have matching initial orientations relative to their respective local gravity vectors. As will be discussed in the next chapter, the target is non-maneuvering and environmental disturbances are not modeled for this work; thus, the target vehicle does not deviate from its initial conditions. The chaser conducts the docking maneuver in such a way that eliminates the relative spacing between the vehicles while causing contact with the target to occur at a low relative velocity in order to avoid damaging either vehicle.

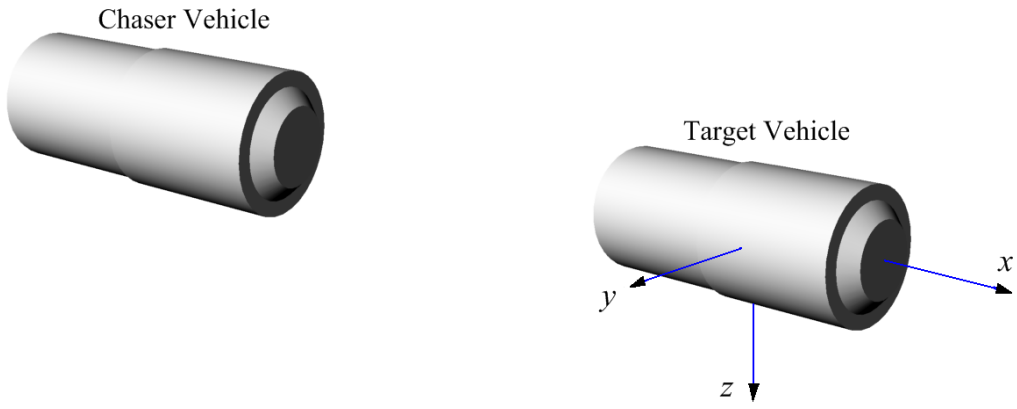

Fig. 7. The Docking Maneuver Coordinate Frame

For this research, the docking maneuver is defined in terms of a target-fixed coordinate system, as shown in Figure 7. This frame is consistent with the body coordinate frame conventions introduced in Chapter II; thus, the frame origin is located at the target center of mass, with x being the along-track coordinate and positive in the direction of orbital motion. The z coordinate is positive radially downward along the local gravity vector, while y completes the right-handed set. Thus, the chaser vehicle begins the docking procedure at some initial position $(-x, 0, 0)$ in docking maneuver coordinates.

Based on these definitions, it should be clear then that the control objective is

to drive the relative position of the chaser vehicle from its initial value of $(-x, 0, 0)$ to the desired final value of $(0, 0, 0)$ relative to the target, or

$$(-x, 0, 0) \ \rightarrow \ (0, 0, 0) \tag{5.1}$$

Additional constraints are also placed on the maneuver to lend realism. First, the total relative velocity magnitude $\sqrt{\left(v_x^2 + v_y^2 + v_z^2\right)}$ at docking is constrained to be sufficiently small to help ensure non-damaging contact between the vehicles, or

$$\sqrt{\left(v_x^2 + v_y^2 + v_z^2\right)} \ < \ \delta \tag{5.2}$$

where δ is a specified velocity threshold. To help minimize the torques on the two vehicles at contact, the final relative attitude of the chaser with respect to the target is also limited. At final docking, the chaser relative attitude in 3-2-1 Euler angles is constrained such that

$$\theta_i \ < \ \kappa_i \tag{5.3}$$

where κ_i is a per-axis attitude angle limit. Thus, a limit is enforced upon each axis simultaneously for the final attitude at docking. Finally, the total maneuver time is required to be such that

$$|(t_{final\ goal} - t_{final})| \ < \ \epsilon \tag{5.4}$$

where ϵ is a specified and small allowed variation in the elapsed time to dock. This ensures that the entire maneuver is conducted in an orderly manner and at a realistic pacing as compared to a real-world docking maneuver. The choices for δ, κ_i, and ϵ for this research will be discussed in Chapter VII.

B. The Sampled-Data Linear Quadratic Regulator

The Linear Quadratic Regulator (LQR) is a type of optimal control method. Optimal controllers $\mathbf{u}(t)$ are controllers that act on a given system in such a way as to minimize a particular cost function chosen by the designer [37, 38]. For example, if one wishes to achieve a particular objective in minimum time, the designer might choose the cost function to be the number of timesteps required to complete the task, or

$$\boldsymbol{J} = \sum_{k=1}^{N} 1 \tag{5.5}$$

as suggested in [37]. LQR belongs to an important category of optimal controllers which have quadratic cost functions, as implied by its name. As also implied by its name, LQR is intended for use on linear, or linearizable, systems. Versions of the LQR exist for the cases in which both the system and the controller are discrete, and in which both the system and controller are continuous; see for example [37] for the derivation of each of these types of LQR controllers.

Sometimes, however, a continuous system must be discretely controlled. This occurs any time a physical (continuous) system is being guided by a controller implemented on a digital (discrete) computer. This situation requires sampling of the continuous system so that the discrete control can be applied, which violates the assumptions underlying both the pure discrete and pure continuous LQR variations. In this case, the proper solution is to employ the SDLQR (LQR for sampled data) control method. The following SDLQR development is drawn primarily from Ref. [39], although both [37] and [38] contain helpful insights as well.

To begin, define a linear system to be the following in continuous-time, state-space form:

$$\dot{\mathbf{x}} = \boldsymbol{A}\mathbf{x} + \boldsymbol{B}\mathbf{u} \tag{5.6}$$

where $\mathbf{x} \in \Re^n$ is the system states vector, $\mathbf{u} \in \Re^m$ represents the control inputs to the system, n is the number of system states, and m is the number of controls. Matrices $\boldsymbol{A} \in \Re^{n \times n}$ and $\boldsymbol{B} \in \Re^{n \times m}$ are the system matrix and the control distribution matrix respectively; they model the nominal dynamics of the system in the presence of a forcing or control input.

Next, an appropriate cost function must be selected. Since all variations of the LQR are regulators, they each seek to drive all system states to zero and hold them there for all time, while using the minimum amount of applied control required to do so. This is a minimum-energy optimization, which can be described in continuous time with the following cost function:

$$\boldsymbol{J} = \frac{1}{2} \int_{t_0}^{t_f} \left(\mathbf{x}^T \boldsymbol{Q}\ \mathbf{x} + \mathbf{u}^T \boldsymbol{R}\ \mathbf{u} \right)\ \mathrm{d}t \tag{5.7}$$

where $\boldsymbol{Q} \in \Re^{n \times n}$ and $\boldsymbol{R} \in \Re^{m \times m}$ are constant weighting matrices selected by the designer such that $\boldsymbol{Q}$ is positive semi-definite and $\boldsymbol{R}$ is positive definite.

For the LQR special case of interest in this work, the SDLQR, it is necessary to convert the problem as currently presented into discrete time. This is because a discrete-time expression for the controller is required for implementation. Refs. [39] and [40] show that a discrete-time sampled data problem formulation can be found that is equivalent to minimizing the continuous cost function from Equation (5.7) subject to the constraint imposed by (5.6). The sampled data discrete equivalent to (5.6) is

$$\mathbf{x}_{k+1} = \boldsymbol{\Phi} \mathbf{x}_k + \boldsymbol{\Gamma} \mathbf{u}_k \tag{5.8}$$

where $\boldsymbol{\Phi}_k \in \Re^{n \times n}$ and $\boldsymbol{\Gamma}_k \in \Re^{n \times m}$ are the discrete analogs of matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ from Equation (5.6); $\boldsymbol{\Phi}_k$ and $\boldsymbol{\Gamma}_k$ are obtained by sampling the continuous system with desired sample period T. The equivalent discrete-time cost function for Equation

(5.7) is

$$\boldsymbol{J} = \frac{1}{2}\sum_{k=1}^{\infty}\left[\mathbf{x}_k{}^T\hat{\boldsymbol{Q}}\ \mathbf{x}_k + \mathbf{u}_k^T\ \hat{\boldsymbol{R}}\ \mathbf{u}_k + 2\left(\mathbf{x}_k{}^T\boldsymbol{M}\mathbf{u}_k\right)\right] \tag{5.9}$$

where matrices $\hat{\boldsymbol{Q}}$, $\hat{\boldsymbol{R}}$, and $\boldsymbol{M}$ are

$$\hat{\boldsymbol{Q}} = \int_0^T \left(\mathrm{e}^{\boldsymbol{A}^T t}\ \boldsymbol{Q}\ \mathrm{e}^{\boldsymbol{A}t}\right)\mathrm{d}t\ , \tag{5.10a}$$

$$\hat{\boldsymbol{R}} = \boldsymbol{R}\cdot T + \boldsymbol{B}^T\left\{\int_0^T\left[\left(\int_0^T \mathrm{e}^{\boldsymbol{A}^T s}\mathrm{d}s\right)\boldsymbol{Q}\left(\int_0^t \mathrm{e}^{\boldsymbol{A}\tau}\mathrm{d}\tau\right)\right]\mathrm{d}t\right\}\boldsymbol{B}\ , \tag{5.10b}$$

and

$$\boldsymbol{M} = \left\{\int_0^T\left[\mathrm{e}^{\boldsymbol{A}^T t}\boldsymbol{Q}\left(\int_0^t \mathrm{e}^{\boldsymbol{A}s}\mathrm{d}s\right)\right]\mathrm{d}t\right\}\boldsymbol{B} \tag{5.10c}$$

respectively. The SDLQR process can now be described as seeking the discrete-time control sequence $\mathbf{u}_k$ that minimizes the discrete cost function (5.9), subject to the constraint (5.8) which is the sampled, discrete-time system dynamical model.

The first step in finding the desired SDLQR optimal control $\mathbf{u}_k$ is to augment the discrete system equation (5.8) into the cost function (5.9). This is done in order to impose the system's characteristics on the cost function minimization process, since the system is a constraint that must be satisfied by any solution obtained. Augmenting the cost function with the system equation is accomplished via Lagrange multiplier theory as presented in [37]. The resulting augmented cost function is

$$\begin{aligned}\boldsymbol{J'} = \sum_{k=1}^{\infty}\Bigg\{\frac{1}{2}\Big[\mathbf{x}_k{}^T\hat{\boldsymbol{Q}}\ \mathbf{x}_k + \mathbf{u}_k^T\ \hat{\boldsymbol{R}}\ \mathbf{u}_k + 2\left(\mathbf{x}_k{}^T\boldsymbol{M}\mathbf{u}_k\right)\Big] \\ + \boldsymbol{\lambda}_{k+1}^T\big(\boldsymbol{\Phi}\mathbf{x}_k + \boldsymbol{\Gamma}\mathbf{u}_k - \mathbf{x}_{k+1}\big)\Bigg\}\end{aligned} \tag{5.11}$$

where $\boldsymbol{\lambda}$ is a Lagrange multiplier [37].

Development of the SDLQR now proceeds through minimization of the augmented cost function. This is accomplished by taking partial derivatives of the $\boldsymbol{J'}$

expression and setting the resulting expressions equal to zero. The partial derivatives of $\boldsymbol{J'}$ with respect to $\boldsymbol{\lambda}_{k+1}$, $\mathbf{x}_k$, and $\mathbf{u}_k$ respectively constitute what are called the 'Necessary Conditions' for a minimum:

$$\frac{\partial \boldsymbol{J'}}{\partial \boldsymbol{\lambda}_{k+1}} = 0 = \boldsymbol{\Phi}\mathbf{x}_k + \boldsymbol{\Gamma}\mathbf{u}_k - \mathbf{x}_{k+1} \tag{5.12a}$$

$$\frac{\partial \boldsymbol{J'}}{\partial \mathbf{x}_k} = 0 = \hat{\boldsymbol{Q}}\mathbf{x}_k + \boldsymbol{M}\mathbf{u}_k + \boldsymbol{\Phi}^T\boldsymbol{\lambda}_{k+1} - \boldsymbol{\lambda}_k \tag{5.12b}$$

$$\frac{\partial \boldsymbol{J'}}{\partial \mathbf{u}_k} = 0 = \hat{\boldsymbol{R}}\mathbf{u}_k + \boldsymbol{M}^T\mathbf{x}_k + \boldsymbol{\Gamma}^T\boldsymbol{\lambda}_{k+1} \tag{5.12c}$$

Notice that the first Necessary Condition directly yields the discrete system equation (5.8), verifying that the system dynamical constraints are indeed being enforced on the minimization process. The third Necessary Condition yields an expression for the desired optimal SDLQR control $\mathbf{u}_k$.

Solving Equation (5.12c), the third Necessary Condition, for $\mathbf{u}_k$ results in

$$\mathbf{u}_k = -\hat{\boldsymbol{R}}^{-1}\left(\boldsymbol{M}^T\mathbf{x}_k + \boldsymbol{\Gamma}^T\boldsymbol{\lambda}_{k+1}\right) \tag{5.13}$$

which is a function of both the state vector $\mathbf{x}_k$ and the Lagrange multiplier $\boldsymbol{\lambda}_{k+1}$. The multiplier can be eliminated from the expression through a change of variables

$$\boldsymbol{\lambda}_k = \boldsymbol{P}_k\mathbf{x}_k \tag{5.14}$$

where $\boldsymbol{P}_k \in \Re^{n\times 1}$ is a symmetric matrix. Substituting this into Equation (5.13) for $\boldsymbol{\lambda}_{k+1}$ yields

$$\mathbf{u}_k = -\hat{\boldsymbol{R}}^{-1}\left(\boldsymbol{M}^T\mathbf{x}_k + \boldsymbol{\Gamma}^T\boldsymbol{P}_{k+1}\mathbf{x}_{k+1}\right) \tag{5.15}$$

It is now necessary to eliminate $\mathbf{x}_{k+1}$ from the expression for $\mathbf{u}_k$, since it is not possible to use the $(k+1)^{th}$ value of the state vector to calculate the control on the

k^{th} timestep. Solving Necessary Condition 1 for $\mathbf{x}_{k+1}$ and substituting gives

$$\mathbf{u}_k \;=\; -\hat{\boldsymbol{R}}^{-1}\left[\boldsymbol{M}^T\mathbf{x}_k + \boldsymbol{\Gamma}^T\boldsymbol{P}_{k+1}\left(\boldsymbol{\Phi}\mathbf{x}_k + \boldsymbol{\Gamma}\mathbf{u}_k\right)\right] \tag{5.16}$$

Next, collect like terms:

$$\left(\boldsymbol{I} + \hat{\boldsymbol{R}}^{-1}\boldsymbol{\Gamma}^T\boldsymbol{P}_{k+1}\boldsymbol{\Gamma}\right)\mathbf{u}_k \;=\; -\hat{\boldsymbol{R}}^{-1}\left(\boldsymbol{M}^T + \boldsymbol{\Gamma}^T\boldsymbol{P}_{k+1}\boldsymbol{\Phi}\right)\mathbf{x}_k \tag{5.17}$$

Pre-multiply both sides by $\hat{\boldsymbol{R}}$ to get

$$\left(\hat{\boldsymbol{R}} + \boldsymbol{\Gamma}^T\boldsymbol{P}_{k+1}\boldsymbol{\Gamma}\right)\mathbf{u}_k \;=\; -\left(\boldsymbol{M}^T + \boldsymbol{\Gamma}^T\boldsymbol{P}_{k+1}\boldsymbol{\Phi}\right)\mathbf{x}_k \tag{5.18}$$

Finally, solve for $\mathbf{u}_k$, obtaining the desired optimal controller:

$$\mathbf{u}_k \;=\; -\boldsymbol{K}_{SDR}\mathbf{x}_k\;, \tag{5.19}$$

where $\boldsymbol{K}_{SDR}$, the optimal controller gain matrix, is defined as

$$\boldsymbol{K}_{SDR} \;=\; \left(\hat{\boldsymbol{R}} + \boldsymbol{\Gamma}^T\boldsymbol{P}_{k+1}\boldsymbol{\Gamma}\right)^{-1}\left(\boldsymbol{M}^T + \boldsymbol{\Gamma}^T\boldsymbol{P}_{k+1}\boldsymbol{\Phi}\right) \tag{5.20}$$

Note that the value for $\boldsymbol{P}_{k+1}$ is required to compute the time-varying value of $\boldsymbol{K}_{SDR}$ at each timestep. $\boldsymbol{P}_{k+1}$ is found by solving an expression known as the matrix Ricatti equation:

$$\boldsymbol{P}_k \;=\; \boldsymbol{\Phi}^T\boldsymbol{P}_{k+1}\boldsymbol{\Phi} + \hat{\boldsymbol{Q}} - \boldsymbol{\chi}^T\left(\hat{\boldsymbol{R}} + \boldsymbol{\Gamma}^T\boldsymbol{P}_{k+1}\boldsymbol{\Gamma}\right)^{-1}\boldsymbol{\chi} \tag{5.21}$$

where $\boldsymbol{\chi}$ is a placeholder variable, defined as

$$\boldsymbol{\chi} \;=\; \boldsymbol{M}^T + \boldsymbol{\Gamma}^T\boldsymbol{P}_{k+1}\boldsymbol{\Phi} \tag{5.22}$$

The matrix Ricatti equation derivation is not shown here; the interested reader should consult [37] or [38] for the full development. The references show that the Ricatti

equation is basically obtained by performing the Lagrange multiplier change of variables from Equation (5.14) on Necessary Condition 2, then substituting Necessary Conditions 1 and 3 into the resulting expression.

As the final time becomes very large so that ($t_{final} \rightarrow \infty$), the time-varying optimal controller gain $\boldsymbol{K}_{SDR}$ converges towards a steady-state value. This means that in situations with very large final times, the steady-state gain value can be computed *a priori* and used for all timesteps, with minimal overall effect on system performance. This is advantageous since it eliminates the need for computing a new gain value at each timestep. The expression for steady-state optimal controller gain is

$$\boldsymbol{K}_{SS} \;=\; \left(\hat{\boldsymbol{R}} + \boldsymbol{\Gamma}^T \boldsymbol{P}_{SS} \boldsymbol{\Gamma}\right)^{-1} \left(\boldsymbol{M}^T + \boldsymbol{\Gamma}^T \boldsymbol{P}_{SS} \boldsymbol{\Phi}\right) \tag{5.23}$$

which is notationally identical to the time-varying expression for $\boldsymbol{K}_{SDR}$ in (5.20); the main difference is that all variables are now constant-valued. Notice that one must still know $\boldsymbol{P}_{SS}$ to find the steady-state optimal gain, analogous to the situation that existed for the time-varying case. As before, the discrete Ricatti equation is required to find $\boldsymbol{P}_{SS}$; the steady-state matrix Ricatti equation is referred to as the algebraic Ricatti equation and is found to be

$$\boldsymbol{P}_{SS} \;=\; \boldsymbol{\Phi}^T \boldsymbol{P}_{SS} \boldsymbol{\Phi} + \hat{\boldsymbol{Q}} - \boldsymbol{\chi}_{SS}^T \left(\hat{\boldsymbol{R}} + \boldsymbol{\Gamma}^T \boldsymbol{P}_{SS} \boldsymbol{\Gamma}\right)^{-1} \boldsymbol{\chi}_{SS} \tag{5.24}$$

where $\boldsymbol{\chi}_{SS}$ is defined as

$$\boldsymbol{\chi}_{SS} \;=\; \boldsymbol{M}^T + \boldsymbol{\Gamma}^T \boldsymbol{P}_{SS} \boldsymbol{\Phi} \tag{5.25}$$

Further discussion of the algebraic Ricatti equation can be found in [38].

C. Controller Implementation

Since the SDLQR design method assumes a linear system model, the nonlinear relative navigation equations had to be linearized before SDLQR could be applied to this research. In Chapter II, the equations governing spacecraft relative position were found to be

$$\dot{x} = u \tag{5.26a}$$

$$\dot{y} = v \tag{5.26b}$$

$$\dot{z} = w \tag{5.26c}$$

$$\dot{u} + \left(\frac{\mu}{{R_c}^3} - n^2\right) x - 2n\dot{z} = \frac{T_x}{m} \tag{5.27a}$$

$$\dot{v} + \left(\frac{\mu}{{R_c}^3}\right) y = \frac{T_y}{m} \tag{5.27b}$$

$$\dot{w} + \left(\frac{\mu}{{R_c}^3} - n^2\right) z + 2n\dot{x} + \mu\left(\frac{{R_c}^3 - R^3}{R^2 {R_c}^3}\right) = \frac{T_z}{m} \tag{5.27c}$$

in first-order form, and the relative orientation equations were

$$\dot{\psi} = \frac{1}{\cos\theta}\left[w_y \sin\phi + w_z \cos\phi\right] \tag{5.28a}$$

$$\dot{\theta} = \frac{1}{\cos\theta}\left[w_y \cos\theta\cos\phi - w_z \cos\theta \sin\phi\right] \tag{5.28b}$$

$$\dot{\phi} = \frac{1}{\cos\theta}\left[w_x \cos\theta + w_y \sin\theta\sin\phi + w_z \sin\theta\cos\phi\right] \tag{5.28c}$$

$$\dot{w_x} = \left(\frac{I_{yy} - I_{zz}}{I_{xx}}\right) w_y w_z + \frac{L_x}{I_{xx}} \tag{5.29a}$$

$$\dot{w_y} = \left(\frac{I_{zz} - I_{xx}}{I_{yy}}\right) w_x w_z + \frac{L_y}{I_{yy}} \tag{5.29b}$$

$$\dot{w_z} = \left(\frac{I_{xx} - I_{yy}}{I_{zz}}\right) w_x w_y + \frac{L_z}{I_{zz}} \tag{5.29c}$$

Please refer to Chapter II for the variable definitions and notation conventions used in these equations. To obtain a linearized version of these equations, perturbation theory and small angle approximations were employed about a nominal operating condition, using a similar procedure to that shown in [41]. The nominal operating condition was selected to be the docked position, so that relative position, velocity, orientation, and angular velocity are all nominally zero. Neglecting higher order terms and enforcing the nominal condition on nominal states resulted in the standard form linear system:

$$\dot{\mathbf{x}} = \boldsymbol{A}\mathbf{x} + \boldsymbol{B}\mathbf{u} \tag{5.30}$$

where

$$\mathbf{x} = \begin{bmatrix} \mathbf{r}_{rel} \\ \theta \\ \dot{\mathbf{r}}_{rel} \\ \dot{\theta} \end{bmatrix} \tag{5.31}$$

with $\mathbf{r}_{rel}$ representing relative position and θ being relative orientation in 3-2-1 Euler angles, so that $\mathbf{x} \in \Re^{12\times 1}$. Vector $\mathbf{u}$ is defined as

$$\mathbf{u} = \begin{bmatrix} \Delta\mathbf{v} \\ \Delta\boldsymbol{\omega} \end{bmatrix} \tag{5.32}$$

where $\Delta\mathbf{v} \in \Re^{3\times 1}$ is the position control vector, and $\Delta\boldsymbol{\omega} \in \Re^{3\times 1}$ is the orientation control vector. System matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \boldsymbol{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{A}_{2,4} \\ \boldsymbol{A}_{3,1} & \mathbf{0} & \boldsymbol{A}_{3,3} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \quad \text{and} \quad \boldsymbol{B} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \boldsymbol{I} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{B}_{4,2} \end{bmatrix} \tag{5.33}$$

with sub-matrices $\boldsymbol{A}_{2,4}$, $\boldsymbol{A}_{3,1}$, $\boldsymbol{A}_{3,3}$, and $\boldsymbol{B}_{4,2}$ defined as follows:

$$\boldsymbol{A}_{2,4} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad \boldsymbol{A}_{3,1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -n^2 & 0 \\ 0 & 0 & 3n^2 \end{bmatrix}$$

$$\boldsymbol{A}_{3,3} = \begin{bmatrix} 0 & 0 & 2n \\ 0 & 0 & 0 \\ -2n & 0 & 0 \end{bmatrix} \qquad \boldsymbol{B}_{4,2} = \begin{bmatrix} \frac{1}{I_{xx}} & 0 & 0 \\ 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix}$$

where $\boldsymbol{I}$ and $\boldsymbol{0}$ are the (3×3) identity matrix and a (3×3) matrix of zeros respectively, and (I_{xx}, I_{yy}, I_{zz}) are the chaser principal moments of inertia in the body frame.

Once a linearized model of the spacecraft docking relative dynamics was obtained, it was necessary to investigate the stability of the resulting linear system. The eigenvalues of the matrix $\boldsymbol{A}$ were found to be as follows:

$$0$$
$$0 + 7.085868940218369e-004i$$
$$0 - 7.085868940218369e-004i$$
$$-1.077425908907470e-018 + 7.085868940218357e-004i$$
$$-1.077425908907470e-018 - 7.085868940218357e-004i$$
$$0$$
$$0$$
$$0$$
$$0$$
$$0$$
$$0$$
$$0$$

These eigenvalues reveal that there are no unstable system modes, several neutrally stable modes, and two stable modes in the system. Thus, the linearized system is stable.

The final step for generating the system model required for using the SDLQR design methodology was to discretize it using a zero-order hold at a sampling rate of $T = 0.5$ seconds. This sampling rate was more than twice the system's fastest response frequency, thus preventing aliasing [38]. Since there was a large final time for this project's docking maneuver, the steady-state SDLQR design procedure could then be employed to create an optimal SDLQR controller for each of the following control paradigms.

1. Controller #1

For the first controller, the Sampled-Data Linear Quadratic Regulator of the previous section was implemented in a fashion patterned after the controller in Ref. [42]. Thus, rather than the vector $\mathbf{u}_k$ consisting of commanded control *magnitudes* as it is in the traditional SDLQR, $\mathbf{u}_k$ was instead specified to be control pulse *lengths*. (To be consistent with this, the chaser propulsion system model for controller #1 was specified to have fixed-output, variable-time, pulsing thrusters.) In this case, the docking controller gain matrix $\boldsymbol{K}_{SS}$ was used to command thruster pulse lengths for each axis of system control. The controller was tuned to dock successfully through an iterative process of adjusting the value of weight matrix $\boldsymbol{Q}$, generating the updated controller gain matrix, and then checking the resulting controller's performance via simulation. The tests performed to evaluate the first controller's performance robustness with respect to different types of disturbances will be detailed in Chapter VIII.

2. Controller #2

For the second controller, the Sampled-Data Linear Quadratic Regulator of the previous section was applied in a more traditional fashion—the vector $\mathbf{u}_k$ consisted of the usual commanded control magnitudes. Thus, the chaser propulsion system model for controller #2 was modified to have fixed-time, variable-output thrusters for translational control, with a set of orthogonal reaction wheels for rotational control. Therefore, vector $\mathbf{u}_k$ was composed of the per-axis commanded total thrust magnitude and the per-axis commanded reaction wheel driver motor torques. (See Chapter VI for more information about the modified chaser actuator system for controller #2.) Just as it was for the other controller, the control gain matrix $\boldsymbol{K}_{SS}$ was tuned through an iterative process of adjusting the value of weight matrix $\boldsymbol{Q}$, generating the updated controller gain matrix, and then checking the resulting controller's performance via simulation. The set of tests used to evaluate this controller's performance robustness with respect to different types of disturbances will also be detailed in Chapter VIII.

CHAPTER VI

AUTOMATED SPACECRAFT DOCKING SIMULATION

The automated docking simulation built to support this work consists of several different parts. Some sections of the simulation were already discussed in preceding chapters: Chapter III described the high-fidelity VisNav sensor model, the Kalman post-filter for relative rate estimates was discussed in Chapter IV, and the docking controller was developed in Chapter V. This chapter will cover the final two parts of the simulation, the chaser and target vehicle models, in Sections A and B respectively. Section C then describes how all of the simulation's parts fit together to form the full medium-fidelity automated spacecraft docking simulation.

A. Chaser Vehicle Model

In order for the simulated docking to be as relevant as reasonably possible, realistic vehicle models were used rather than simplified or generic vehicle models. The realistic models were derived from a currently operational automated rendezvous and docking vehicle actually being used in the "real world"—the Automated Transfer Vehicle (ATV) introduced in Chapter I. As mentioned there, the ATV is used by the European Space Agency (ESA) as an automated re-supply shuttle for the International Space Station (ISS). The vehicle was chosen for this project because some public data was available for it, which aided in the creation of a moderately realistic ATV vehicle model. The ATV was chosen rather than the H-II Transfer Vehicle (HTV), another vehicle currently under development (and also mentioned in Chapter I), primarily because of the different methods of docking the vehicles use. The ATV performs a direct dock like the Russian Progress vehicle, but the HTV stops near the ISS, is grasped by a robotic arm, and is brought in for the final dock under human

control. Therefore, since the ATV docking method is more closely aligned with the interests of the current research project, the ATV was chosen as the basis for the simulated vehicle models instead of the HTV.

1. Model Properties

Figure 8 shows an artist's rendering of the ATV from the ESA website [3]. As seen in the figure, the ATV is a largely cylindrical vehicle with solar panel 'wings' extending from each side of the vehicle in an X configuration. The dimensions and mass data for the ATV were freely available from ESA, but the principle moments of inertia were not. Therefore, the inertias were estimated by assuming the ATV was a cylindrical rigid body with uniform density and constant cross-section. Note that this assumption omitted the moments of inertia of the solar panels, so the resulting inertias are only a first-order estimate of the real vehicle's inertias.

Fig. 8. The Automated Transfer Vehicle (an artist's rendering) [3]

Arriving at a value for the chaser model's nominal mass also required some

assumptions. First, the vehicle was assumed to be fully loaded with cargo for this project, as it would be to fly a re-supply mission. However, the vehicle was assumed to have used some of its on-board fuel during rendezvous and final approach. Thus, the total (fully loaded) ATV vehicle mass provided by ESA was reduced by an amount considered to be representative of the fuel spent in maneuvering.

To simplify the modeling complexity for this project, vehicle mass and moments of inertia were all assumed to be constant during automated docking maneuvers. The constant mass assumption implies that the amount of fuel burned for the simulation maneuvers is negligible. This is an acceptable approximation because the propulsion system uses relatively low thrust values in comparison to the total vehicle mass. Assuming constant inertias implies that the vehicle is a perfectly rigid body, which is a common first-order modeling approximation. A summary of vehicle data for the real ATV and the assumed values used in this research's models are presented in Table II.

Table II. ATV and Vehicle Model Characteristics

Property	**ATV** [3]	**Chaser Model**	**Target Model**
Mass, kg	20750 (at launch)	19000 (at dock)	
Length, m	10.3	10.594 (with docking collars)	
Diameter, m	4.48 max	4.48 max	4.48 max
Inertia, $kg \cdot m^2$	N/A	$I_{xx} = 47667$, $I_{yy} = I_{zz} = 191810$	
Propulsion	4 thrusters @ 490 N, 28 @ 220 N		none

2. Chaser Equations of Motion

The next step in generating a chaser vehicle model was to determine the equations of motion (EOM) that would describe it dynamically. Assuming two-body motion, that the vehicle was affected only by the gravity of the body it was orbiting and no other objects, the EOM could be de-coupled into separate nonlinear translational and rotational equations [43]. This assumption is valid to several decimal places of accuracy while greatly reducing the complexity of the resulting equations. Thus it is an assumption that is commonly made in spacecraft simulations. From Ref. [43], but expressed in first-order vector form here instead of second-order vector form, the inertial translational EOM for the chaser vehicle are:

$$\dot{\boldsymbol{R}}_c = \boldsymbol{V}_c \tag{6.1a}$$

$$\dot{\boldsymbol{V}}_c = -\frac{\mu \boldsymbol{R}_c}{\left(\boldsymbol{R}_c^T \boldsymbol{R}_c\right)^{3/2}} + \frac{\boldsymbol{T}_c}{m} \tag{6.1b}$$

The c subscript designates 'chaser' vehicle quantities, μ is the gravitational constant of the body being orbited, m is the mass of the chaser vehicle, $\boldsymbol{T}_c$ is the inertial thrust vector produced by the chaser, $\boldsymbol{R}_c$ is the inertial position vector of the vehicle, and $\boldsymbol{V}_c$ is the inertial velocity vector of the vehicle.

The chaser vehicle's rotational EOM were affected by the choice of coordinate frame conventions used in this work. Since the relative motion between the chaser and target vehicles were described using 3-2-1 Euler orientation angles as discussed in Chapter II, 3-2-1 Euler angles were also used to describe the inertial vehicle orientations for consistency. The chaser vehicle's moments of inertia were also affected by the choice of coordinate frames, because the body-fixed coordinate frame convention specified in Chapter II was aligned with the principal body axes. This caused the

chaser inertia matrix to be diagonal [18]:

$$\boldsymbol{I} = \begin{bmatrix} I_{11} & I_{12} & I_{13} \\ I_{21} & I_{22} & I_{23} \\ I_{31} & I_{32} & I_{33} \end{bmatrix} \quad \Longrightarrow \quad \boldsymbol{I}_p = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{6.2}$$

The assumed values of the principal inertias I_{xx}, I_{yy}, and I_{zz} for the chaser vehicle are listed in Table II. Based on the coordinate frame choices for this project, the so-called Euler rotational EOM had the following form [18] for the chaser vehicle model:

$$\dot{\Psi} = \frac{1}{\cos\Theta}\left[w_y \sin\Phi + w_z \cos\Phi\right] \tag{6.3a}$$

$$\dot{\Theta} = \frac{1}{\cos\Theta}\left[w_y \cos\Theta\cos\Phi - w_z \cos\Theta\sin\Phi\right] \tag{6.3b}$$

$$\dot{\Phi} = \frac{1}{\cos\Theta}\left[w_x \cos\Theta + w_y \sin\Theta\sin\Phi + w_z \sin\Theta\cos\Phi\right] \tag{6.3c}$$

$$\dot{w_x} = \left(\frac{I_{yy} - I_{zz}}{I_{xx}}\right) w_y w_z + \frac{L_x}{I_{xx}} \tag{6.4a}$$

$$\dot{w_y} = \left(\frac{I_{zz} - I_{xx}}{I_{yy}}\right) w_x w_z + \frac{L_y}{I_{yy}} \tag{6.4b}$$

$$\dot{w_z} = \left(\frac{I_{xx} - I_{yy}}{I_{zz}}\right) w_x w_y + \frac{L_z}{I_{zz}} \tag{6.4c}$$

Here, (Ψ, Θ, Φ) are the 3-2-1 Euler angles (yaw, pitch, and roll) describing the chaser vehicle's orientation relative to the inertial reference frame; (w_x, w_y, w_z) are the chaser's angular velocities expressed in the body-fixed (chaser) frame; (I_{xx}, I_{yy}, I_{zz}) are the principal inertias in the body frame; and (L_x, L_y, L_z) are the body frame components of the chaser's applied torque vector.

The chaser vehicle's EOM contain both thrust and torque terms because the chaser is assumed to be maneuverable. The chaser is the only one of the two vehicles that maneuvers during the docking; the target is assumed to be in a 'parked' mode for the duration of the maneuver. However, only limited information about the

ATV's propulsion system could be found. This lack of information required several assumptions by the author to facilitate the creation of a useable chaser propulsion system model. The next sub-section will describe the resulting model and how it was generated.

3. Chaser Propulsion System

Several assumptions were required when generating a viable chaser vehicle propulsion model for this project since detailed publicly available information about the ATV's propulsion system was unavailable. The same number of thrusters with the same thrust magnitudes were used as the ATV employs, but a thruster configuration had to be created due to lack of information about the real vehicle's propulsion system. Also, since no details of the ATV Attitude Control System (ACS) were found, the chaser model's thrusters were made simultaneously responsible for both translation and orientation control. This decision greatly increased the thruster firing logic complexity, but it also eliminated having to model a system for which there was no available information.

A series of assumptions were made to reduce propulsion system modeling complexity while attempting to retain at least some measure of realism. All chaser vehicle thrusters and engines were assumed to fire at fixed thrust values, in a bang-off-bang operational concept. No intermediate values of thrust were available from any thruster. Next, the thruster fuel supply was assumed to be infinite. Therefore, regardless of the length or complexity of the maneuver commanded, the vehicle was assumed to always have enough fuel on-board to complete it. Fuel consumption was also not modeled, so that the weight of the chaser remained constant throughout the docking maneuver as previously discussed. The thrusters themselves were largely assumed to be perfect: first, the thrusters were capable of updating themselves as

rapidly as the docking controller could command them. Next, there was never any variation in the magnitude or direction of thruster firings. Finally, the thrusters were assumed to never fail, weaken, or wear out—they would always fire when commanded, for as long as commanded, at their maximum thrust rating. The only major element of realism modeled for the thrusters was actuator lag, so that the thrusters always trailed the on/off commands received from the controller by a set interval of time.

The first task of the propulsion system model was to convert controller commands into a set of firing commands for specific thrusters. The model decided what thrusters to turn on and for how long in order to best satisfy the thrust pulse commands received from the controller (see Chapter V for information about the controller's operation). The propulsion system model also checked for redundant thruster firings. These would occur in two ways: (1) when two thrusters directly opposite each other and pointed in opposite directions were both simultaneously commanded to fire to satisfy given controller commands, and (2) when one thruster was commanded to fire for both position and orientation control at the same time. In case (1), the opposite firings effectively cancelled out and had no net effect on the vehicle, so both thrusters were turned off to save fuel. For case (2), the propulsion system model ignored the shorter of the two commanded thruster pulses and executed the longer one. After eliminating redundant thruster firings, the propulsion system model then passed along the simplified commands to the thrusters for execution. This logic was automatically applied every time the controller specified a thruster firing.

The propulsion model was not particularly high-fidelity. However, enforcing some rudimentary firing logic and a specific thruster configuration onto the controller did place some real-world-type constraints on its performance. This ensured at least a basic level of realism for the experimental tests conducted, since the controller was not allowed to respond in an ideal and uninhibited way to the modeling errors it

encountered.

4. Chaser Model Modifications for Controller #2

The second controller evaluated in this research required some modifications to the just-described chaser vehicle model. While the assumed mass properties remained the same, the methods of actuation of the vehicle were altered rather significantly. First, the thrusters were modified to be fixed-time, variable-output rather than variable-time, fixed-output. As was discussed in Chapter V, this allowed a more traditional application of the SDLQR control methodology than was used for controller #1. Because of this change to variable-output thrusters, one additional constraint was placed on the thrusters to prevent chattering—that of a minimum thrust level. The controller had to specify a commanded torque of at least 5% of a thruster's max output before it would fire. All other parts of the thruster modeling were left unchanged: there was still a maximum allowed output for each thruster, redundant thrusters were checked for and eliminated, and actuator lag was modeled as a zero-order hold.

The other actuator modification for the controller #2 chaser model was the introduction of reaction wheels for rotational control. Reaction wheels are often used in the space field, particularly on smaller vehicles and satellites, due to their relative simplicity as control actuators. A reaction wheel consists of a rapidly spinning disk that is mounted in a fixed position on a vehicle's structure [32]. Typically, they are used in sets of at least three (more if redundancy is desired), with each one having a spin axis co-aligned with one of the vehicle's principal axes. In order to exert control over the vehicle, the spin rate of a reaction wheel is altered by applying torque to it via an electric motor. Thus the motor either speeds the reaction wheel up or slows it down as desired, and the corresponding change in momentum of the spinning reaction wheel exerts a net torque on the vehicle to alter its angular rate. The three reaction

wheels used in the chaser model for controller #2 were scaled-up, simplified models of a line of reaction wheels manufactured by Honeywell for use on small satellites [44].

The addition of the reaction wheels for attitude control required some modifications to the rotational equations of motion for the chaser vehicle model, though the translational EOM remained unchanged. Since the controller was designed to provide commanded reaction wheel motor torques, the altered rotational EOM were modeled after those from [45] and are as follows in first-order form:

$$\dot{\Psi} = \frac{1}{\cos\Theta}\left[w_y \sin\Phi + w_z \cos\Phi\right] \tag{6.5a}$$

$$\dot{\Theta} = \frac{1}{\cos\Theta}\left[w_y \cos\Theta\cos\Phi - w_z \cos\Theta\sin\Phi\right] \tag{6.5b}$$

$$\dot{\Phi} = \frac{1}{\cos\Theta}\left[w_x \cos\Theta + w_y \sin\Theta\sin\Phi + w_z \sin\Theta\cos\Phi\right] \tag{6.5c}$$

$$\dot{w_x} = \frac{(I'_{yy} - I'_{zz})w_y w_z + J_2 w_z(w_y + \nu_2) - J_3 w_y(w_z + \nu_3) + M_1}{I'_{xx}} \tag{6.6a}$$

$$\dot{w_y} = \frac{(I'_{zz} - I'_{xx})w_z w_x + J_3 w_x(w_z + \nu_3) - J_1 w_z(w_x + \nu_1) + M_2}{I'_{yy}} \tag{6.6b}$$

$$\dot{w_z} = \frac{(I'_{xx} - I'_{yy})w_x w_y + J_1 w_y(w_x + \nu_1) - J_2 w_x(w_y + \nu_2) + M_3}{I'_{zz}} \tag{6.6c}$$

$$\dot{\nu_1} = -\dot{w_x} - \frac{M_1}{J_1} \tag{6.7a}$$

$$\dot{\nu_2} = -\dot{w_y} - \frac{M_2}{J_2} \tag{6.7b}$$

$$\dot{\nu_3} = -\dot{w_z} - \frac{M_3}{J_3} \tag{6.7c}$$

Where (Ψ, Θ, Φ) are the 3-2-1 Euler angles (yaw, pitch, and roll) describing the chaser vehicle's orientation relative to the inertial reference frame; (w_x, w_y, w_z) are the chaser's angular velocities expressed in the body-fixed (chaser) frame; (ν_1, ν_2, ν_3) are the reaction wheel angular velocities expressed in the same frame; $\left(I'_{xx}, I'_{yy}, I'_{zz}\right)$

are the augmented chaser principal inertias in the body frame where $I'_{xx} = (I_{xx} - J_1)$ and so forth; and (M_1, M_2, M_3) are the applied reaction wheel drive motor torques.

The addition of reaction wheels to the chaser model for rotation control certainly increased the complexity of both the rotationl EOM used and the controller tuning process required to accomodate them. However, it had the benefit of completely decoupling the translational and rotational portions of the control problem, thanks to the particular set of thruster locations used (which was the same for both controllers). This was because the chosen thruster set contributed no net torques on the chaser vehicle when the translation control jets were active, regardless of which pairs of thrusters were firing. So, the reaction wheels could operate upon the vehicle's attitude state unimpeded, while the thrusters no longer had to balance control requests between two sometimes conflicting interests and could be used solely for translation.

B. Target Vehicle Model

The model for the target vehicle was desired to be as relevant to automated docking as reasonably possible, just as the chaser vehicle model was. Therefore, the target model was derived from the ATV as well. The chaser and target vehicle models are nearly identical, except that the target is assumed to be non-maneuverable during the docking scenario. Thus, a target vehicle propulsion model was unnecessary and therefore omitted, making the target vehicle much easier to simulate than the chaser while still being a relevant dynamical model for an automated docking scenario.

For simplicity, it was assumed that both the nominal mass and nominal inertias of the target were identical to those of the chaser. They were all assumed to be constant for the duration of the simulation docking maneuver as well, just as they were for the chaser. This resulted in a rigid body target vehicle model, which is a

common first-order modeling assumption. It is notable that the target vehicle mass really was constant during docking, unlike the chaser vehicle, since the target was non-maneuvering and not expending fuel. Thus, the constant mass assumption actually was more valid for the target model than it was for the chaser. The mass, dimensions, and assumed inertias for the target vehicle model are summarized in Table II, along with the real ATV's and the chaser vehicle model's characteristics.

Since the target vehicle model was dynamically identical to the chaser model, the equations of motion differ only by the omission of the propulsion terms. Thus, the inertial translational EOM for the target vehicle in first-order vector form are [43]:

$$\dot{\boldsymbol{R}}_t = \boldsymbol{V}_t \tag{6.8a}$$

$$\dot{\boldsymbol{V}}_t = -\frac{\mu \boldsymbol{R}_t}{\left(\boldsymbol{R}_t^T \boldsymbol{R}_t\right)^{3/2}} \tag{6.8b}$$

where the t subscript designates 'target' vehicle quantities, μ is the gravitational constant of the body being orbited, $\boldsymbol{R}_t$ is the inertial position vector of the vehicle, and $\boldsymbol{V}_t$ is the inertial velocity vector of the vehicle.

The choices of principal axes for the target body frame and 3-2-1 Euler angles for expressing the target vehicle's inertial orientation are identical to the conventions used in generating the chaser vehicle model. Thus, the rotational EOM for the target vehicle model [18] are also identical to those of the chaser, except for the omitted propulsion terms:

$$\dot{\Psi} = \frac{1}{\cos\Theta}\left[w_y \sin\Phi + w_z \cos\Phi\right] \tag{6.9a}$$

$$\dot{\Theta} = \frac{1}{\cos\Theta}\left[w_y \cos\Theta\cos\Phi - w_z \cos\Theta\sin\Phi\right] \tag{6.9b}$$

$$\dot{\Phi} = \frac{1}{\cos\Theta}\left[w_x \cos\Theta + w_y \sin\Theta\sin\Phi + w_z \sin\Theta\cos\Phi\right] \tag{6.9c}$$

$$\dot{w}_x = \left(\frac{I_{yy} - I_{zz}}{I_{xx}}\right) w_y w_z \tag{6.10a}$$

$$\dot{w}_y = \left(\frac{I_{zz} - I_{xx}}{I_{yy}}\right) w_x w_z \tag{6.10b}$$

$$\dot{w}_z = \left(\frac{I_{xx} - I_{yy}}{I_{zz}}\right) w_x w_y \tag{6.10c}$$

where (Ψ, Θ, Φ) are the inertial 3-2-1 Euler orientation angles (yaw, pitch, and roll) of the target vehicle, (w_x, w_y, w_z) are the target vehicle's angular velocities expressed in the target body frame, and (I_{xx}, I_{yy}, I_{zz}) are the vehicle's principal inertias in the body frame.

C. Simulation Development

This section describes how all of the parts of the simulation fit together and are integrated to form the complete automated docking scenario simulation. The entire simulation was developed in Matlab^{TM}; this was to allow direct use of the pre-existing high-fidelity VisNav sensor model described in Chapter III. A top-level schematic of the docking simulation architecture is shown in Figure 9.

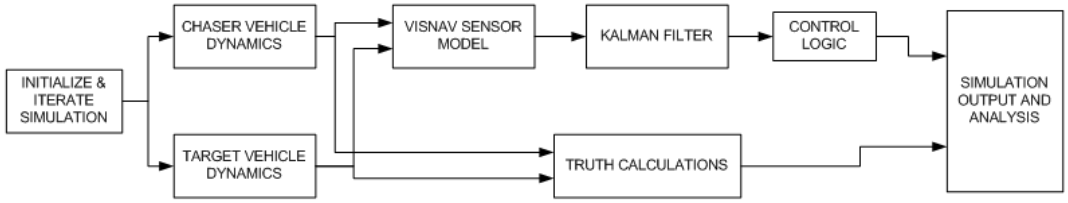

Fig. 9. Docking Simulation Architecture

As shown in the figure, the code simulates two different vehicles in the same circular orbit around the same planetary body. The chaser is initially some user-determined distance behind the target vehicle in the orbit, because the simulation assumes that rendezvous and final approach have already occured to obtain the starting

distance specified by the user. The VisNav sensor model simulates a relative navigation system on-board the chaser vehicle, while the Kalman filter generates real-time estimates of the relative velocity and angular velocity between the vehicles from the VisNav position and orientation estimates. The controller guides the chaser vehicle model to perform a simulated automated docking maneuver based on the VisNav and Kalman filter estimates. As this is happening, the simulation separately calculates truth values at every timestep for all vehicle states and relative states. This set of truth calculations allows for comparative analysis between the estimates from the VisNav-Kalman filter system and what was really happening during a simulation run. Various results graphs can then be plotted and all data (both truth and estimates) can be saved for future analysis.

The automated docking simulation is a useful tool for docking and proximity operations analysis. Its partially modular construction could allow for substitution of different vehicle models, different relative navigation sensor systems, and even of different controllers as the user desires. The simulation can be easily reconfigured to accomodate orbits of any size and inclination, about any planetary body for which the user has characteristic gravitational data. Currently, the simulation's major limitation is that it contains no modeling of on-orbit disturbances such as gravity-gradient torques, atmospheric drag, solar radiation pressure, etc. This limits the realism of its simulated docking scenarios, especially for low altitude orbits. However, even in these cases the simulation can be a useful first-order modeling tool. With further development to allow modeling of environmental disturbances, this simulation tool could be a valuable platform for future research into spacecraft docking and proximity operations.

CHAPTER VII

EXPERIMENT DESIGN

This chapter defines the experiment used to evaluate the performance of the automated docking system formulated in previous chapters. The docking system, which includes a docking controller, the VisNav relative navigation sensor, and the integrated Kalman filter velocity & angular rate estimator, was analyzed through a series of tests. The tests were performed using the docking simulation described in Chapter VI as the experimental testbed. First, the docking system was exercised in nominal conditions, absent any other sources of error aside from the VisNav and Kalman filter sensor noise. These nominal runs established a baseline of comparison for further parametric tests. Additional test cases were then performed to evaluate the nominal system's closed-loop docking performance under four types of modeling uncertainty: chaser vehicle mass modeling errors, uncertainty in the chaser vehicle moments of inertia, variation in the docking scenario's relative starting position, and variation in the chaser's initial attitude relative to the target vehicle. The questions answered by the experiment were:

1. What are the docking system's precision and accuracy characteristics in nominal operating conditions?

2. Does docking controller performance degrade due to vehicle modeling uncertainties, such as mass errors or errors in the moments of inertia?

3. How robust is the controller with respect to initial condition errors such as variation in the relative starting position or relative initial attitude of the chaser?

4. If the controller's performance robustness is less than desired for the types of uncertainty considered, then what modifications should be made to the controller

to improve its performance robustness?

The automated docking scenario used for all of the experimental test cases is presented in Section A, along with docking success criteria and other considerations. Section B describes the docking controller tuning criteria. Finally, Section C defines the specific test cases used in the experiment to evaluate each of the two controllers considered.

A. Automated Spacecraft Docking Scenario

The automated docking scenario used for this experiment was in lunar orbit. The baseline orbit was circular, at an altitude of 400 kilometers and an inclination angle of 30 degrees. The simulation contained only a first-order approximation of the orbital environment, so no environmental disturbances were modeled. Therefore, the orbit did not vary or degrade over time. The target and chaser vehicles were both initially placed in the same orbit, with the chaser assigned to a nominal relative starting position 50 meters behind the target vehicle. The target vehicle was non-maneuvering and therefore never deviated from the baseline orbit. The chaser was commanded by the docking controller to maneuver in order to achieve a successful docking if it was possible to do so. For each docking attempt, the simulation ended either when the along-track (x) distance between the vehicles reached zero, or when the total distance between the vehicles became less than one centimeter. The final conditions at simulation conclusion were then evaluated to determine the success or failure of the attempt.

Docking success or failure was based on four factors: (1) the final relative position error between the two vehicles, (2) the final relative velocity between them, (3) the final relative attitude between them, and (4) the total time required for the maneuver.

These factors were evaluated as follows:

- The final total position error was calculated using the expression

$$\text{position error} \;=\; \sqrt{x^2 + y^2 + z^2} \tag{7.1}$$

  in the coordinate system that Chapter V established for the docking scenario. For controller #1, the total position error had to be less than twenty centimeters for a docking attempt to be a success. An electromagnetic docking mechanism was assumed, so the mechanism was considered capable of drawing the chaser vehicle the rest of the way into contact with the target at that distance based on the performance of the Russian Progress docking system [7]. For controller #2, the total position error was required to be less than ten centimeters for a docking attempt to be successful. This was well within the capabilities of the assumed Progress-like docking system.

- To prevent damage to either vehicle at contact, the final relative velocity was constrained below a threshold value δ as discussed in Chapter V. The required value of δ was chosen to be ten centimeters per second, so that:

$$\sqrt{\left(v_x^2 + v_y^2 + v_z^2\right)} \;<\; \delta \;;\; \delta \;=\; 10.0 \text{ cm/s} \tag{7.2}$$

  where v_x, v_y, and v_z were the relative velocity components of the chaser vehicle with respect to the target vehicle. Therefore, final relative velocity magnitude had to be slower than ten centimeters per second for a docking attempt to be considered a minimum success. However, note that a "preferred" criteria value of five centimeters per second was also established for the final velocity constraint; an attempt improving upon this preferred value was considered to have excelled at this criteria.

- To help minimize the torques on the two vehicles at contact, the final relative attitude of the chaser with respect to the target was constrained. At final docking, the chaser relative attitude in 3-2-1 Euler angles was constrained to a maximum value of 2.0 degrees on each axis:

$$\theta_i \; < \; \kappa_i; \; \kappa_i \; = \; 2.0 \text{ deg/s} \tag{7.3}$$

  Just as for final velocity, a preferred criteria was also established for the final attitude; a run that excelled at this criteria would need to have met a constraint of only 1.0 degree on each axis.

- Total docking maneuver time was constrained to ensure that the entire scenario was conducted at a realistic pace compared to real-world docking maneuvers. For realism, the target maneuver completion time was chosen to be ten minutes. As specified in Chapter V, the maneuver time was allowed to vary from the target completion time by an amount ϵ; ϵ was selected to be 30 seconds:

$$|(t_{final\ goal} - t_{final})| \; < \; \epsilon \; ; \; \epsilon \; = \; 30.0 \text{ seconds} \tag{7.4}$$

  Therefore, a docking attempt had to take at least nine and one-half minutes to complete, while not requiring more than ten and one-half minutes to finish, for the attempt to be considered a success.

B. Docking Controller Tuning Criteria

The docking controller was tuned by iteratively adjusting the weighting matrix $\boldsymbol{Q}$ introduced in Section B of Chapter V. The matrix was adjusted based on the time-domain performance of the closed-loop docking system so that it could meet the previous section's criteria for a successful dock 100% of the time in nominal conditions.

Once a candidate tuning was found, its ability to meet the 100% nominal success rate requirement was verified through extensive simulation. The tuning that gave the best overall performance in nominal conditions was then chosen for use in the experimental tests.

C. Closed-Loop Evaluation of Controller Performance

Table III. Simulation Test Cases, Controller #1

Test Case	**Property Tested**	**Distribution**	**Variation from Nominal**
Case 1	Chaser mass	Uniform	max = ±15%
Case 2	Chaser inertias	Uniform	max = ±15%
Case 3	Start Position	Gaussian	$\sigma = 5\%$

The performance of the automated docking system using each controller was evaluated via a series of simulation runs under different conditions. For both controllers, the scenario described in Section A was first run under nominal conditions to build a basis of comparison for the other test cases. Then, the simulation was run through the off-nominal cases to see how well the docking system handled that particular type of disturbance.

For controller #1, each of the test cases, including the nominal case, was run twenty times apiece. At initialization for each of the off-nominal runs, a value for the property of interest was selected using a pseudo-random number generator adjusted to the proper range and distribution of values for that test case. The run was then conducted and its results were recorded so that performance comparisons could be made when all tests concluded. A summary of the off-nominal cases used for eval-

uation of controller #1 is provided in Table III. Note that controller #1 was not evaluated against an initial relative attitude variation case, which is why such a case does not appear in the table.

For controller #2, the evaluation was performed with "extremal" cases only; after a candidate tuning was verified to work repeatably for the nominal case, it was subjected to a series of runs that evaluated its performance against only the maximum and minimum values used for each off-nominal case for controller #1. However, controller #2 was also subjected to test cases covering initial relative attitude variation, unlike controller #1. A summary of the twelve test cases that were used to evaluate controller #2 is provided in Table IV.

The final experimental results for both controllers will be presented in the next chapter.

Table IV. Simulation Test Cases, Controller #2

Test Case	**Property Tested**	**Variation from Nominal**
Case 1	Start Position	$+15\%$
Case 2	Start Position	-15%
Case 3	Chaser inertias	$I_{xx} = +15\%, I_{yy} = +15\%, I_{zz} = +15\%$
Case 4	Chaser inertias	$I_{xx} = -15\%, I_{yy} = -15\%, I_{zz} = -15\%$
Case 5	Chaser inertias	$I_{xx} = +15\%, I_{yy} = -15\%, I_{zz} = -15\%$
Case 6	Chaser inertias	$I_{xx} = -15\%, I_{yy} = +15\%, I_{zz} = +15\%$
Case 7	Init. Rel. Attitude	$\Psi = +7.5^\circ, \Theta = +7.5^\circ, \Phi = +7.5^\circ$
Case 8	Init. Rel. Attitude	$\Psi = -7.5^\circ, \Theta = -7.5^\circ, \Phi = -7.5^\circ$
Case 9	Init. Rel. Attitude	$\Psi = +7.5^\circ, \Theta = -7.5^\circ, \Phi = -7.5^\circ$
Case 10	Init. Rel. Attitude	$\Psi = -7.5^\circ, \Theta = +7.5^\circ, \Phi = +7.5^\circ$
Case 11	Chaser mass	$+15\%$
Case 12	Chaser mass	-15%

CHAPTER VIII

NUMERICAL EXAMPLES

The performance of the spacecraft docking controllers described in Chapter V was evaluated through a series of tests. This was accomplished by implementing each controller in the automated docking simulation from Chapter VI, then putting each one through the experimental test plan assigned to it as defined in the previous chapter. The results of these tests are presented here.

The organization of this chapter is as follows: the results from controller #1 are presented in Section A, with the results from each test case performed on it presented in sub-sections. The overall results for controller #1 are summarized in the final sub-section of Section A. The results from controller #2 are presented in Section B, with the results of all of its test cases also in sub-sections. The summary of the controller #2 results are presented in the final sub-section of Section B.

A. Results for Controller #1

1. Test Case 1: Nominal Conditions

The nominal docking case is an automated spacecraft docking maneuver performed with sensor noise being the only disturbance present. The sensor noise is the result of the VisNav sensor model and the Kalman filter velocity estimator being included in the simulation loop. All chaser vehicle properties such as mass and moments of inertia, as well as relative starting position of the maneuver with respect to the target vehicle, retain their nominal values for the duration of each nominal simulation run.

An example of the relative trajectory flown by the chaser during a successful controller #1, Test Case 1 simulation run is shown in Figure 10. The trajectory is

expressed in target frame coordinates, in accordance with the convention chosen to describe the automated docking maneuver for this work in Chapter V. The chaser vehicle is initially at position $(-50, 0, 0)$ meters in this coordinate system, and it steadily closes the distance to the target vehicle's constant position near $(0, 0, 0)$ meters.

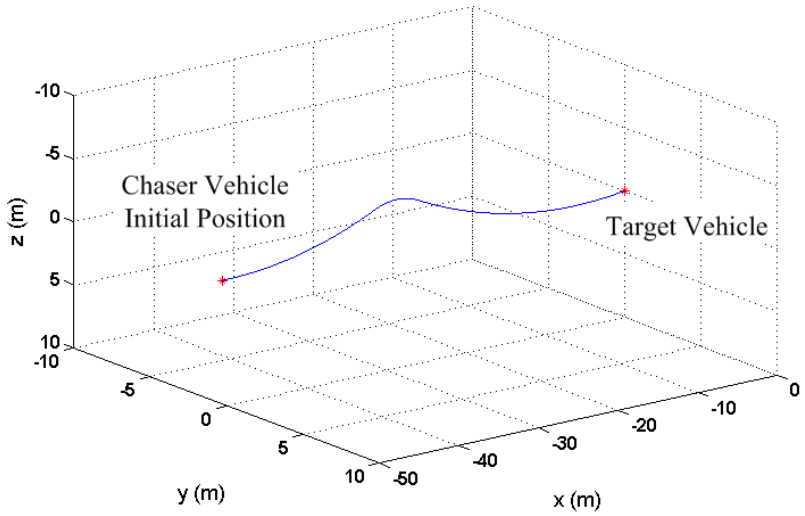

Fig. 10. Docking Maneuver Relative Trajectory, Test Case 1

The chaser vehicle's relative position, orientation, and rate states during the example Test Case 1 docking maneuver are shown in Figure 11. Note that the vehicle moves steadily towards the target in the along-track or x direction, while drifting slightly towards negative z. Once the z-direction drift reaches a critical threshold, the propulsion system reacts to correct it back toward zero as the maneuver continues. Throughout the maneuver, even during the z correction, the controller successfully maintains the orientation and angular velocity states very near their initial values of zero. Finally, the position states all converge to zero as well and docking occurs at 601.56 seconds of elapsed simulation time, with a final total position error of less than

1.0 centimeter.

Figure 12 shows a plot of the magnitude of the chaser vehicle's total relative velocity vector versus time for the Test Case 1 example maneuver. Inspection of the figure shows that the total velocity magnitude was well-constrained throughout the maneuver, ensuring that the vehicle proceeded at a suitably slow velocity at all times. The final velocity magnitude is approximately 8.8 centimeters per second, verifying that the relative velocity at contact between the vehicles occurred slower than the desired velocity limit of 10 centimeters per second.

The chaser vehicle's propulsion system activity during the Test Case 1 example is depicted in Figure 13. As the figure illustrates, the beginning of the maneuver features a period of heavy activity for the x-axis thrusters, causing the vehicle to begin closing the along-track distance to the target. Once sufficient forward momentum has been accrued to achieve the maneuver time performance criteria, while not exceeding the maximum velocity criteria, the x thrusters cease their activity and the vehicle coasts for a time. Then, as the vehicle begins to drift off-track in the z-direction, the z-axis thrusters fire briefly to correct its course. Finally, the z thrusters shut off, and the vehicle completes the maneuver by coasting to the target. It is interesting to note that y thruster firings are never required at any time during this particular docking scenario.

An enlarged view of the thruster firing history plot is provided in Figure 14. This image illustrates the pulsed nature of the propulsion system's applied thrust. Recall from Chapter V that the control system operates by specifying thruster pulse lengths. The propulsion system then translates these commanded pulse lengths into thruster activity as described in Chapter VI. This results in the actual behavior depicted in the figure, where the thrusters fire only intermittently while resting the majority of the time. It should be noted that while only x-axis thruster activity has been enlarged

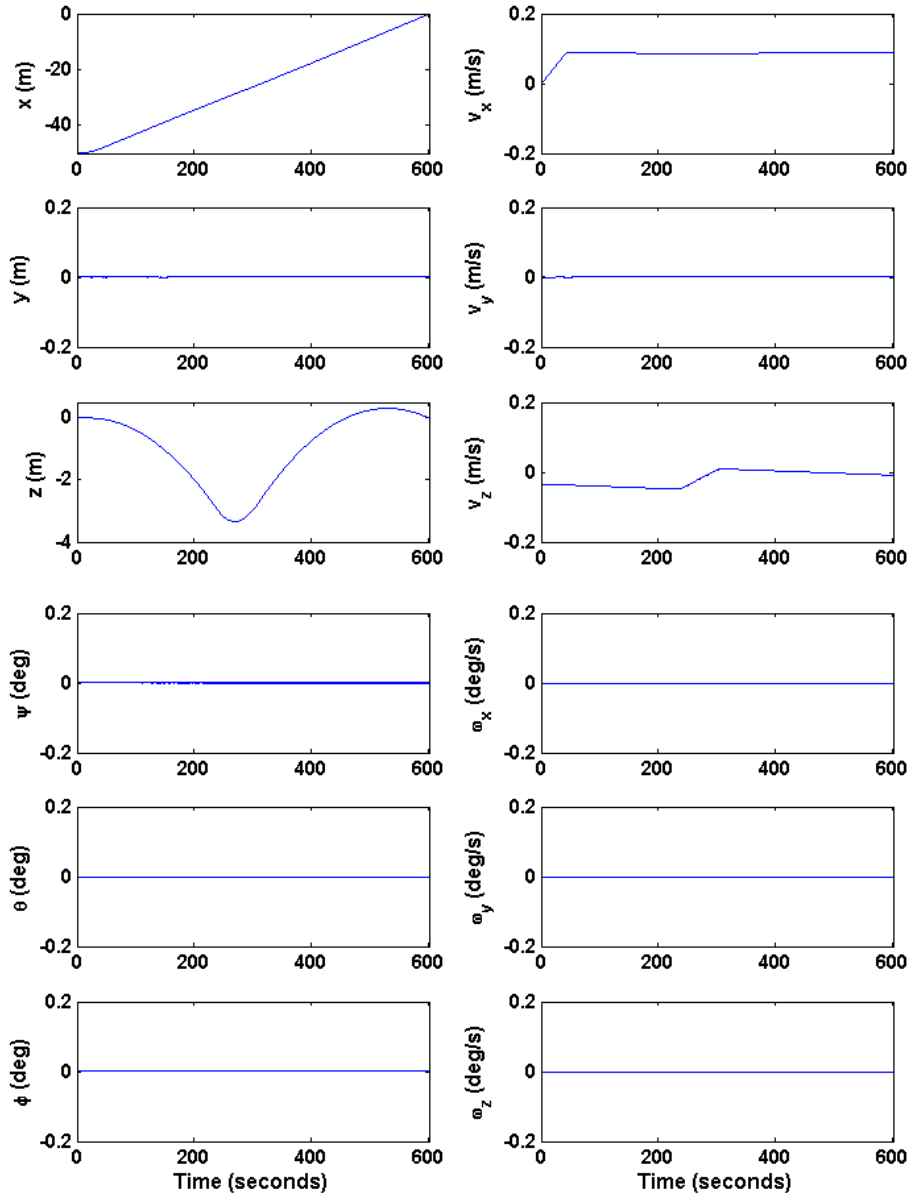

Fig. 11. Chaser Vehicle Relative States, Test Case 1

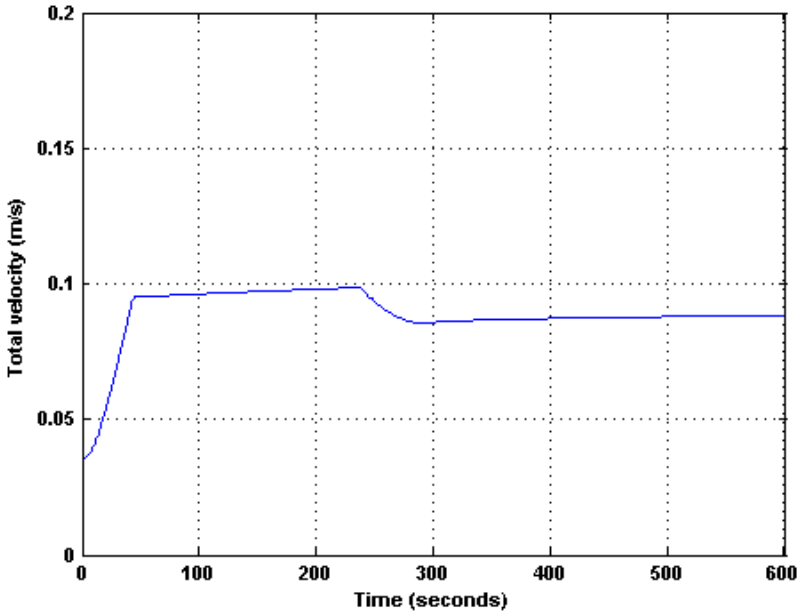

Fig. 12. Total Relative Velocity Profile, Test Case 1

for brevity, the same pulsing behavior occurs in all thrust system axes.

Figure 15 presents the relative navigation estimation accuracy of the VisNav sensor and integrated Kalman filter during the example Test Case 1 dock. The estimate errors for all position and orientation states are quite large initially, even lying far outside the 3-σ error bounds for each state. However, at close range the estimation errors all converge neatly within the 3-σ error bounds (except for Ψ, the relative yaw angle). This estimate error behavior is primarily due to the VisNav sensor's calibration being range-dependent. Using the current calibration methods, the technician must choose which part of the sensor's operating range the calibration will optimize. When forced to choose, it makes sense to specify the greatest sensor accuracy at close range for docking applications. Thus, VisNav is currently calibrated so that its best performance during docking maneuvers is at an operating range of about 10 meters or less.

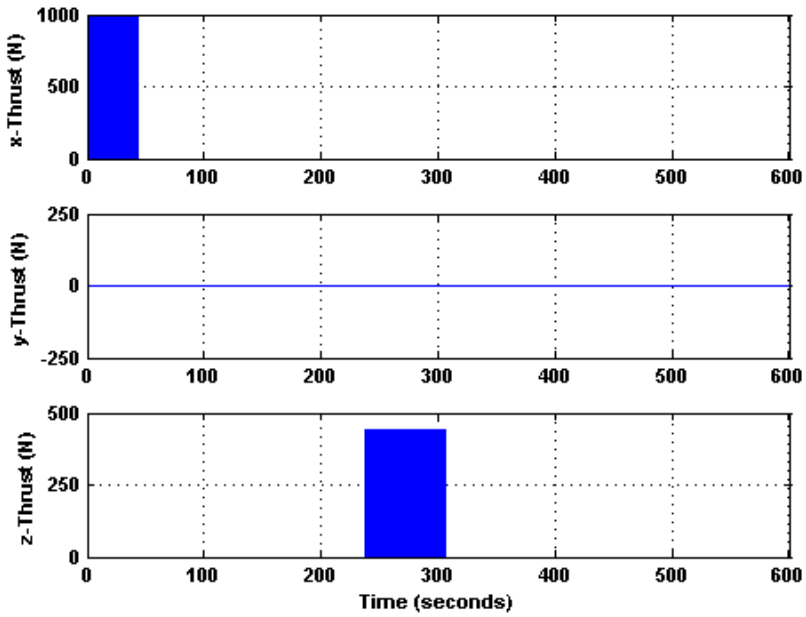

Fig. 13. Chaser Vehicle Thrust Profile, Test Case 1

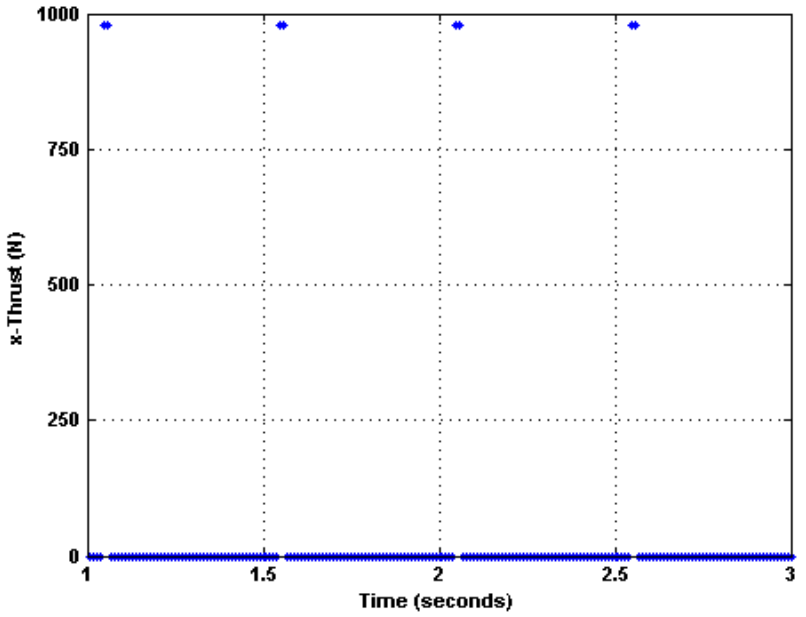

Fig. 14. Chaser Vehicle Thrust Enlarged, Test Case 1

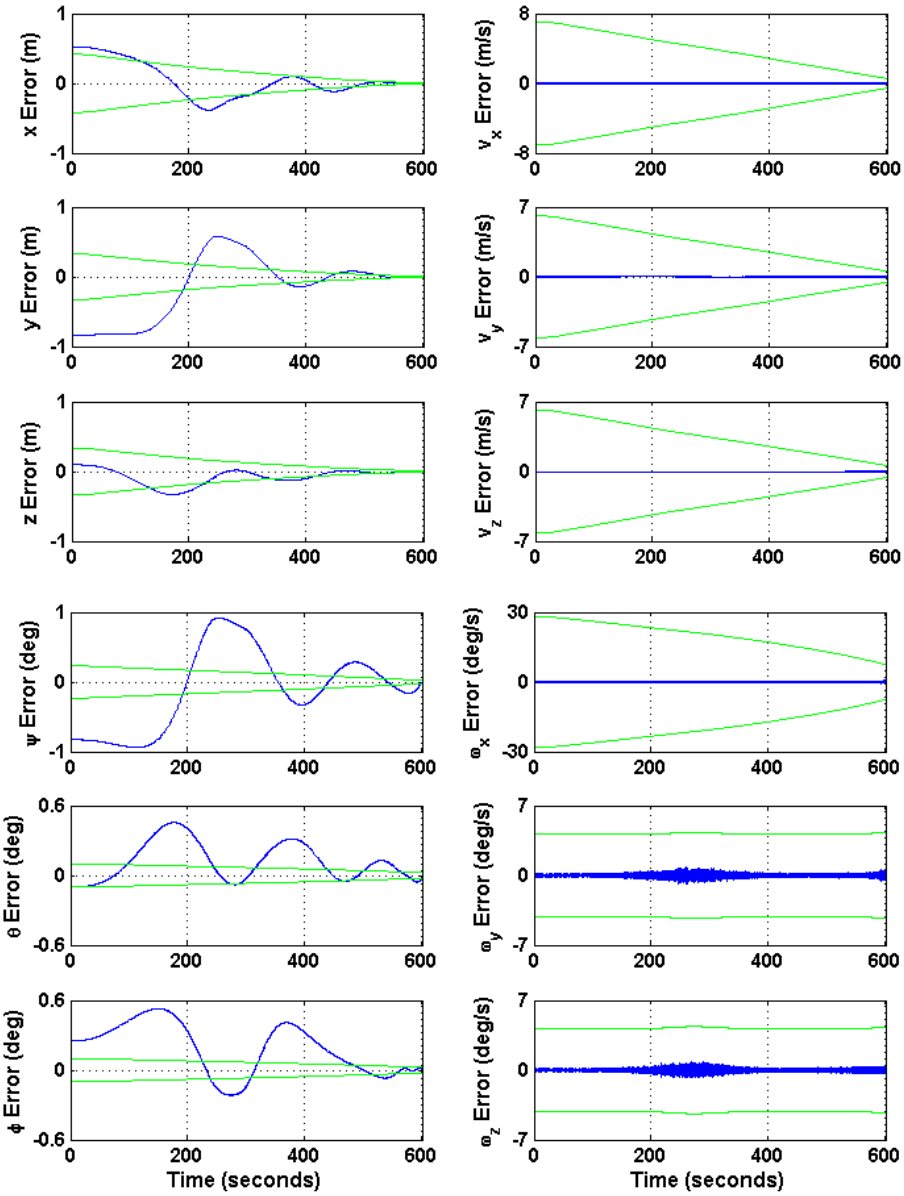

Fig. 15. VisNav & Kalman Filter Estimation Error, Test Case 1

Note how the velocity 3-σ error bounds all appear to be much wider than necessary. This is a result of the tuning chosen for the Kalman filter. In an effort to mitigate the effects of the range dependent calibration on the estimate accuracies, the filter was tuned to increase its error bounds in the position and orientation states. A side effect of this is that it was very conservative in assigning the velocity state error bounds, resulting in the bounds being overly wide. However, when comparing the actual estimates to the real velocity and angular velocity values, it is interesting that its estimates are actually far better than the filter itself considered them to be.

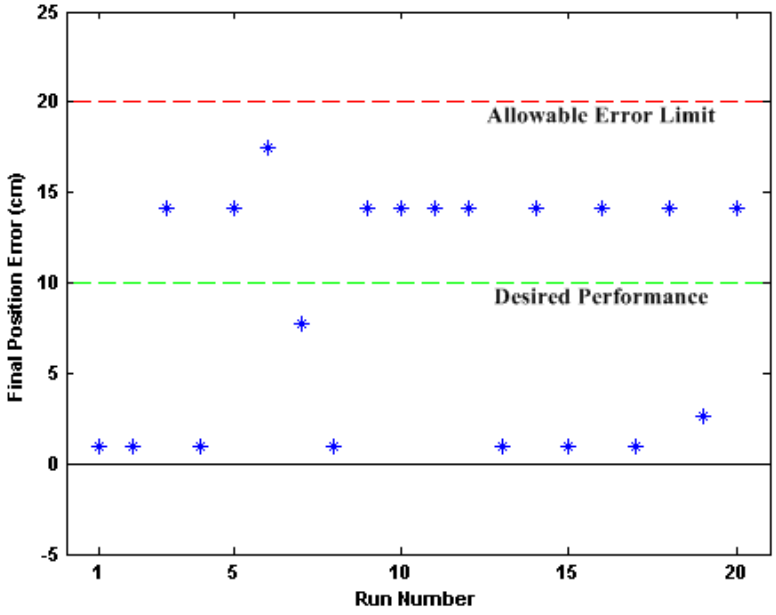

Fig. 16. Final Position Error Results Summary, Test Case 1

Overall, the performance of VisNav and the integrated Kalman filter was deemed satisfactory in nominal conditions. While the position and orientation errors were large initially, the estimates converged very nicely at close range so that the controller was able to affect docking without difficulty. Also, the velocity estimates were very accurate in spite of the Kalman filter being almost overly conservative; this too

enabled successful docking thanks to reliable information.

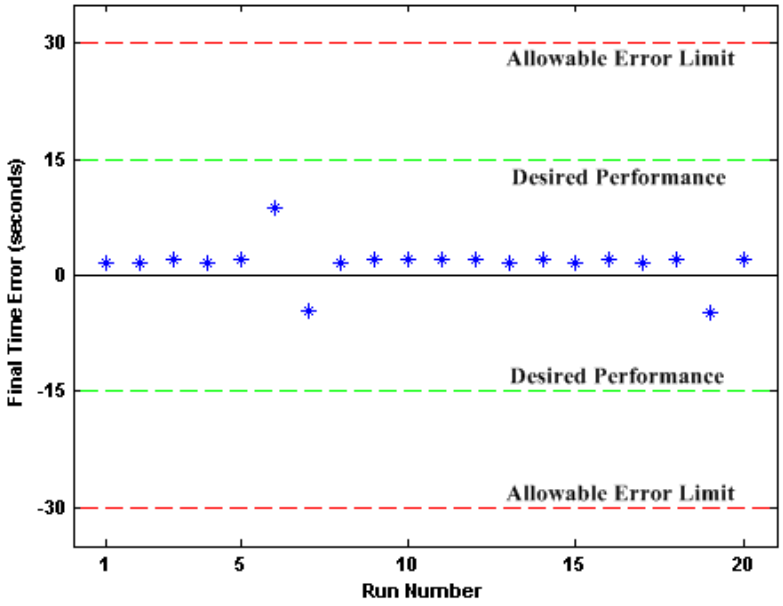

Fig. 17. Final Time Error Results Summary, Test Case 1

As specified in the experimental test plan developed in Chapter VII, twenty simulation runs were performed for Test Case 1 to build a baseline for comparison with the other test cases. The results of the nominal test runs are shown in Figures 16, 17, and 18, which summarize how each of the twenty runs performed with respect to the design criteria in final position error, final time error, and final total velocity respectively. Notice that all twenty runs are considered successful docks because they meet the minimum design criteria. Nine of the runs also meet the additional 'preferred' standards in final position error and final time error. These results will be further analyzed in the context of the experimental results for controller #1 as a whole in Sub-section 5.

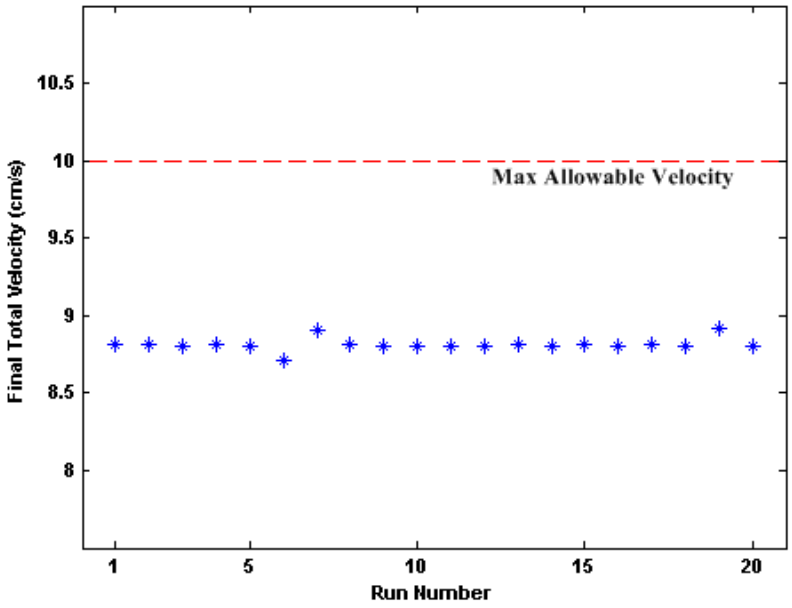

Fig. 18. Final Total Velocity Results Summary, Test Case 1

2. Test Case 2: Chaser Mass Variation

Test Case 2 for controller #1 involved performing the same automated spacecraft docking maneuver as for the nominal case, except that chaser vehicle mass modeling uncertainty was introduced into the scenario, in addition to the sensor noise that was present in Test Case 1. As prescribed in Chapter VII, the mass of the chaser for each Test Case 2 run was randomly varied by up to ±15% of the nominal value according to a uniform distribution. At the beginning of each test run, a value for the chaser mass was generated; the selected value was then held constant for the entirety of that run. All other scenario parameters such as chaser vehicle moments of inertia and relative starting position of the maneuver retained their nominal values in Case 2 tests.

An example relative trajectory from a successful Test Case 2 simulation run is shown in Figure 19. The trajectory is again expressed in target frame coordinates

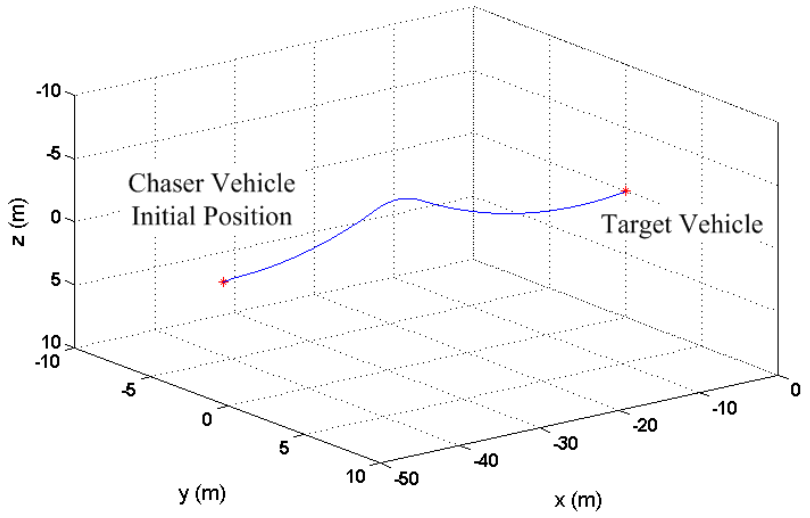

Fig. 19. Docking Relative Trajectory, Test Case 2

as Chapter V conventions specify. As in Test Case 1, the chaser vehicle begins the docking maneuver at $(-50, 0, 0)$ meters relative to the target vehicle, and concludes its motion with a successful dock to the target vehicle.

Figure 20 presents the chaser vehicle relative state information from the example Test Case 2 docking maneuver. The graphs appear nearly identical to those for the nominal case, with similar behavior in all states. The same drifting phenomenon again occurs in the z-position state as the vehicle proceeds towards the dock, until the propulsion system again corrects it back to zero with a small amount of overshoot. Eventually, all of the states converge to zero at the same time, causing a successful dock to occur at simulation time 600.97 seconds with a final position error of about 5 centimeters.

The time history of the chaser vehicle's total relative velocity magnitude for the example Test Case 2 dock is shown in Figure 21. This graph also appears very similar to the one for the nominal docking maneuver; the velocity was well constrained as

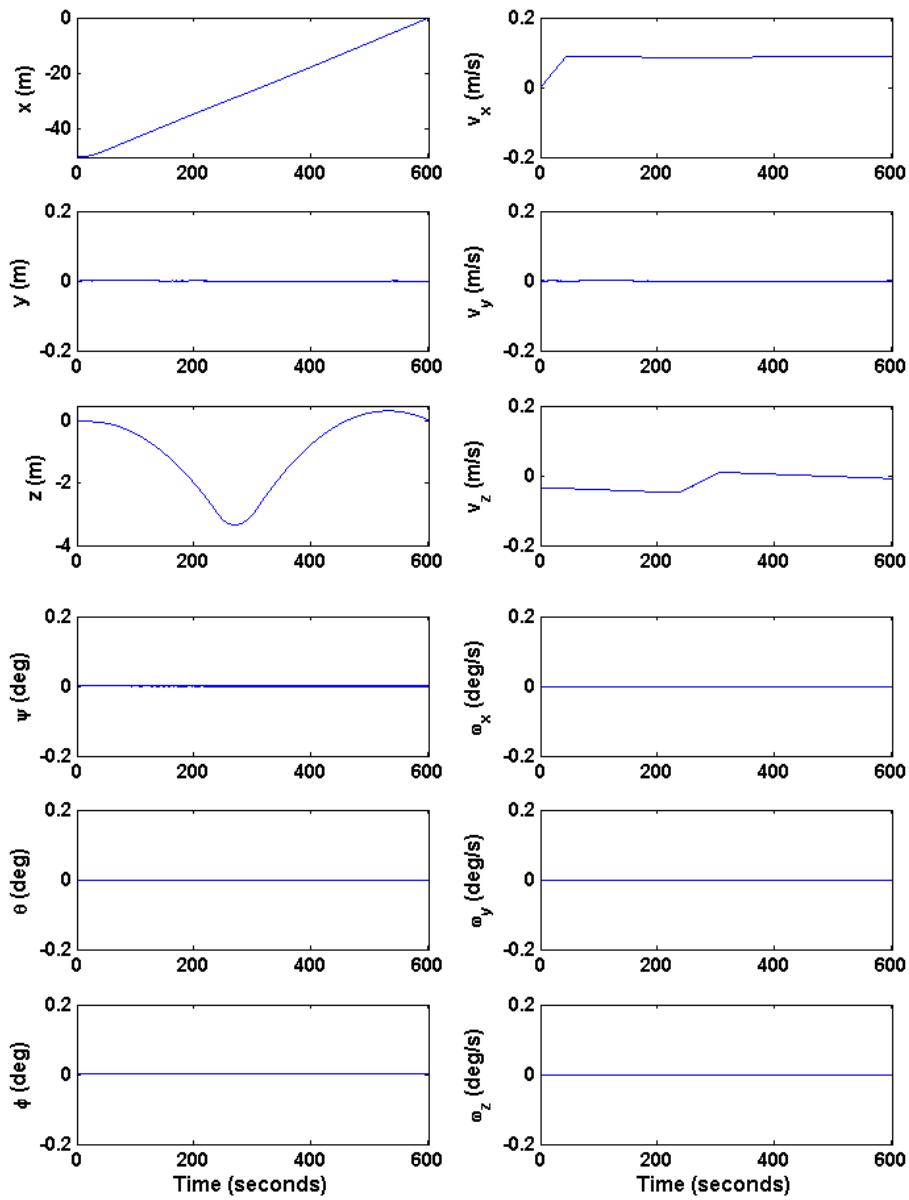

Fig. 20. Chaser Relative States, Test Case 2

for the nominal case, and the final velocity at dock was again approximately 8.8 centimeters per second. Since the desired velocity limit for successful dock was 10 centimeters per second for all cases, this example Case 2 run met the final velocity criteria for a successful dock.

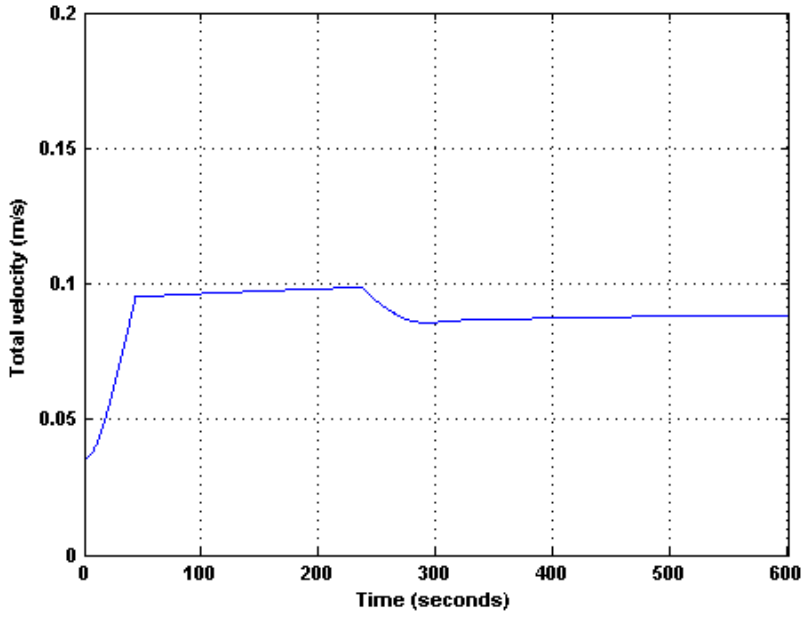

Fig. 21. Total Relative Velocity, Test Case 2

Figure 22 illustrates the chaser vehicle propulsion system activity that resulted in the successful dock in Test Case 2. As seen in the figure, the maneuver features a similar thrust profile as for Test Case 1: a initial period of heavy activity for the x-axis thrusters, followed by a long coast with no propulsion; a brief z-axis thruster firing to correct vehicle drift in the middle of the maneuver; and a second coast to the completion of the maneuver. Again, the y-axis thrusters are never used during the Test Case 2 example dock. Also, the thruster firings shown are still of a pulsed nature like they were for the nominal case, in which the thrusters were inactive most of the time while firing only intermittently. Thus, the maneuver was very fuel efficient

while still satisfactorily accomodating an off-nominal chaser mass value with nearly 200 kilograms error in its value (nominal is 19,000 kg, while the run shown used a chaser mass value of 19,194 kg).

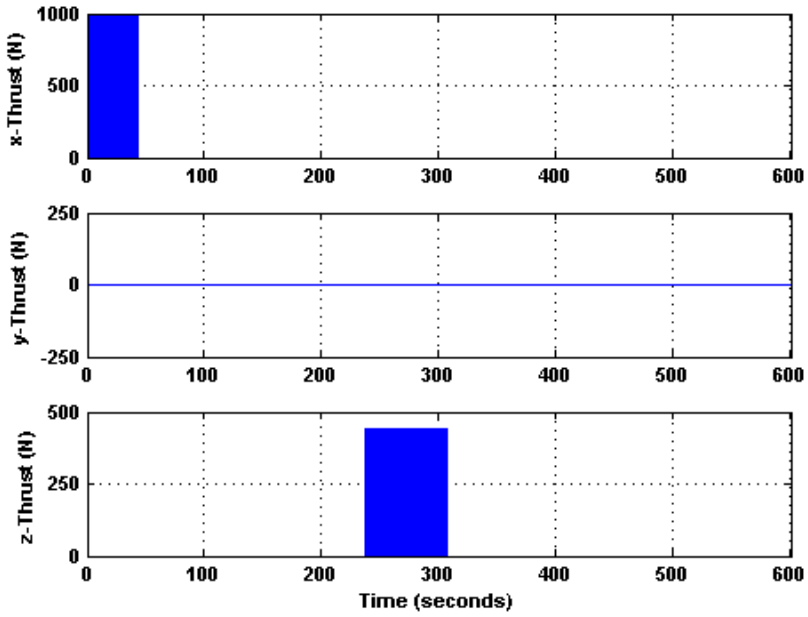

Fig. 22. Chaser Thrust Profile, Test Case 2

The VisNav sensor and integrated Kalman filter estimator performance for Test Case 2 is presented in Figure 23. Similar to the Test Case 1 example, the estimate errors for all position and orientation states are initially far outside the 3-σ error bounds for each state. However, the near-range errors again generally converge to well within the 3-σ error bounds except for relative yaw angle. Thus, the range-dependent VisNav sensor calibration affected the estimates for Test Case 2 in a similar way as it did the nominal case estimates. This makes sense based on how similar the trajectory flight paths are for the two cases, and it suggests that the calibration range dependency effects on the estimate errors might be predictable and consistent even in the presence of relative dynamics modeling uncertainty.

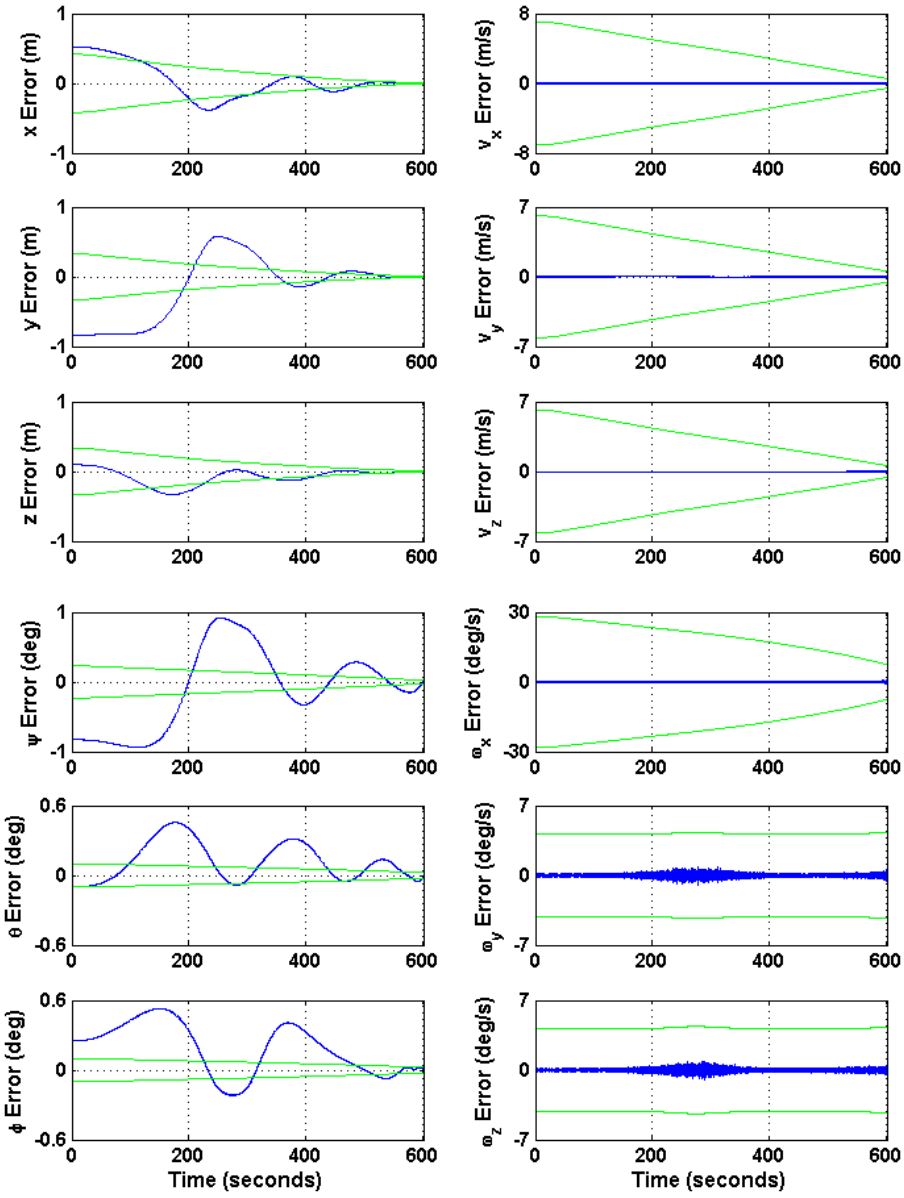

Fig. 23. VisNav & Kalman Filter Estimation Error, Test Case 2

Overall, VisNav and the integrated Kalman filter exhibited satisfactory performance in Test Case 2 simulation runs. The position and orientation estimates converged from their large initial errors to much more acceptable errors at close range, enabling the controller to effect docking without difficulty. In addition, though the velocity estimates from the Kalman filter were again overly conservative as they were in nominal conditions, they still proved to be very accurate by comparison to the truth values. This too helped to facilitate successful Test Case 2 docking.

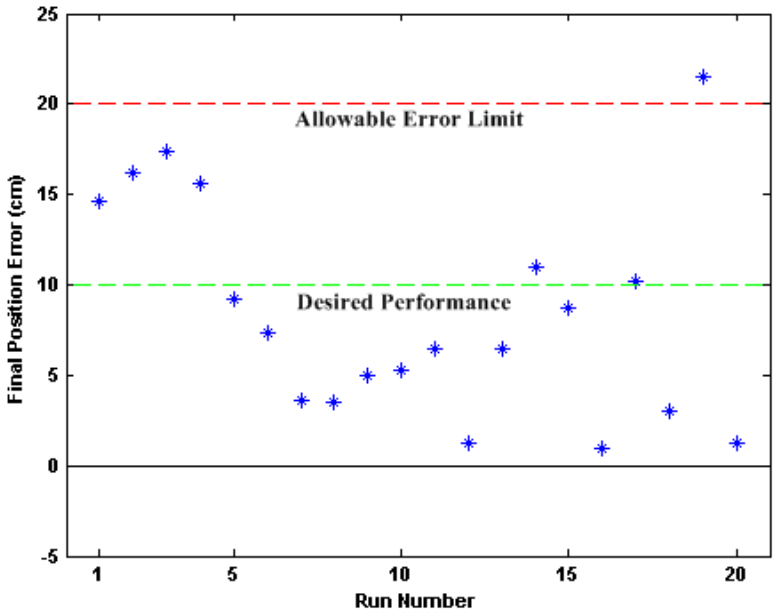

Fig. 24. Final Position Error Results Summary, Test Case 2

Twenty simulation runs were performed for Test Case 2 in order to evaluate the docking controller's performance robustness with respect to chaser mass modeling uncertainty. The results of these runs are summarized in Figures 24, 25, and 26, which illustrate how each of the runs performed with respect to the design criteria in final position error, final time error, and final total velocity respectively. Mass uncertainty was found to affect docking reliability somewhat, though not to an insurmountable

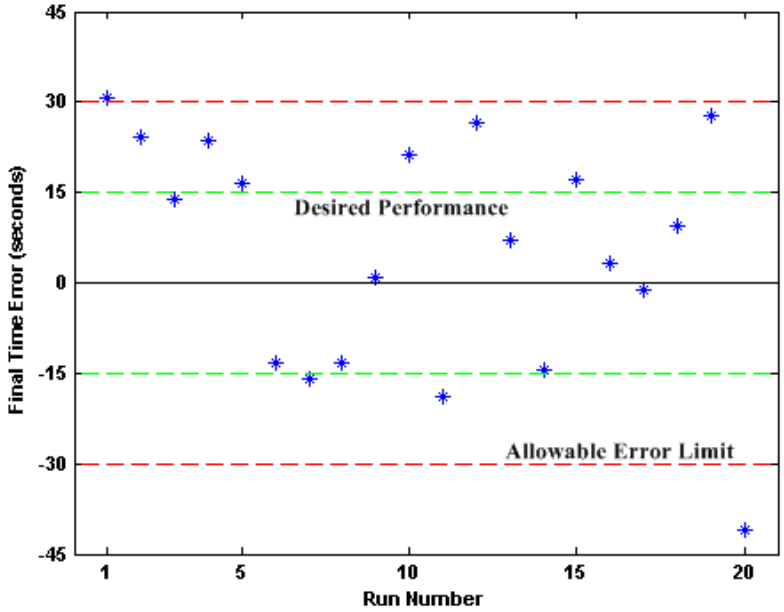

Fig. 25. Final Time Error Results Summary, Test Case 2

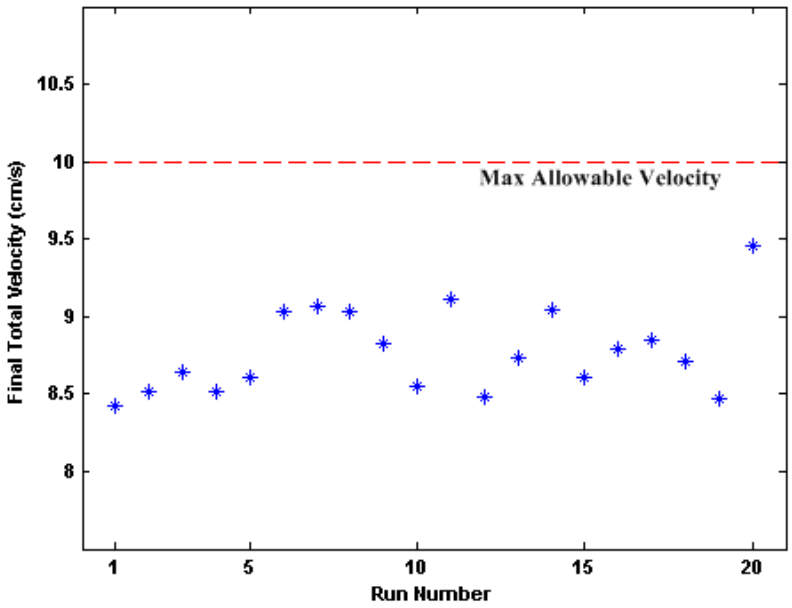

Fig. 26. Final Total Velocity Results Summary, Test Case 2

extent, as evidenced by seventeen of the twenty runs resulting in successful docks. In fact, the three unsuccessful runs all had chaser mass values near the extremes of the mass uncertainty range tested, yet they all still achieved success according to two out of three categories each. This is an encouraging result. Sub-section 5 will contain further analysis of the Test Case 2 results in the context of the entire controller #1 portion of the experiment.

3. Test Case 3: Chaser Inertias Variation

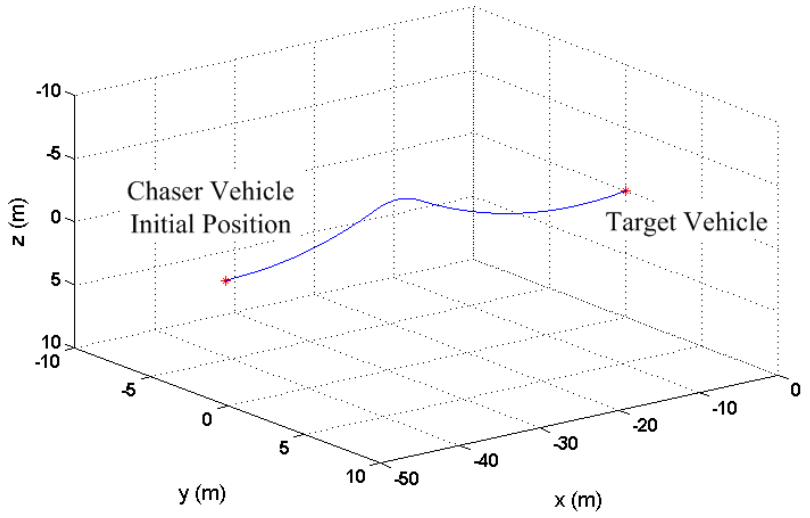

Fig. 27. Docking Relative Trajectory, Test Case 3

Controller #1 Test Case 3 evaluated the docking controller's robustness to chaser vehicle moments of inertia modeling uncertainty in the presence of sensor noise. The scenario used was the same as the nominal case, except for the values of the moments of inertia. The three principal chaser moments of inertia were each randomly varied by up to $\pm 15\%$ of their nominal values for each simulation run as prescribed in the experimental test plan presented in Chapter VII. At the beginning of each test run, values for each of the moments of inertia were individually selected according to a

uniform distribution. The selected values were each then held constant for the entirety of that run. All other scenario parameters such as chaser vehicle mass and maneuver relative starting position retained their nominal values for all Case 3 tests.

The relative trajectory traversed by the chaser vehicle from an example Test Case 3 simulation run is presented in Figure 27; the trajectory is in target frame coordinates as was standard for this project. Since relative starting position was held constant at its nominal value for this case, the maneuver began at $(-50, 0, 0)$ meters relative to the target vehicle and concluded in a successful dock when contact was made with the target vehicle near $(0, 0, 0)$ meters.

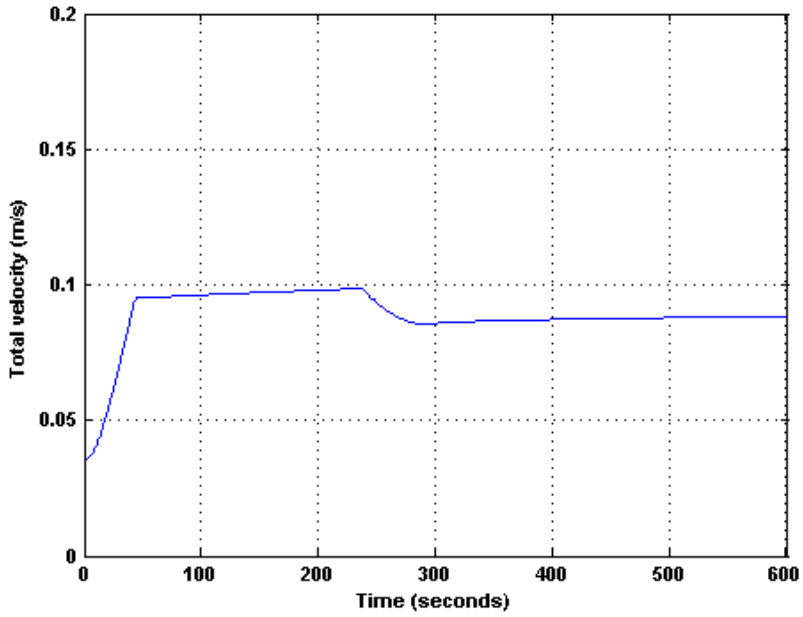

Fig. 28. Total Relative Velocity, Test Case 3

Figure 28 shows a plot of the chaser vehicle's total relative velocity magnitude versus time for the example Test Case 3 dock. It is evident by inspection that the total velocity had similar general behavior in this example as it did for both the nominal case and the mass variation case. The total velocity remained slower than the

maximum allowable final velocity at all times, including when contact between the vehicles occurred. The total relative contact velocity was approximately 8.8 centimeters per second, just as it was for the previous two cases. Thus, the example Test Case 3 run shown here met the final velocity criteria for a successful dock.

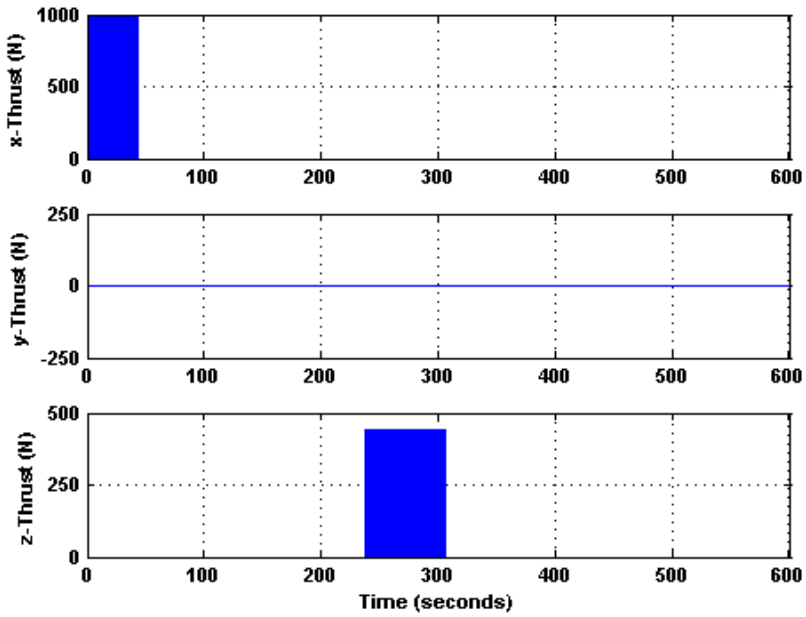

Fig. 29. Chaser Thrust Profile, Test Case 3

The chaser vehicle thruster activity during the Test Case 3 example maneuver is shown in Figure 29. As expected after seeing the total relative velocity plot, the thrust profile is similar to the nominal and mass variation cases. It features a similar initial period of heavy x-axis thruster activity, followed by a relatively long period of coasting, then a brief z-axis thruster firing to correct vehicle drift, and a final coast until contact is made with the target vehicle to complete the maneuver. Notice that the y-axis thrusters are again never used during this example dock, just as for the first two cases. Although it is not quite clear in the figure, the thruster firings shown are of a pulsed nature just as the others were; the thrusters fired only intermittently while

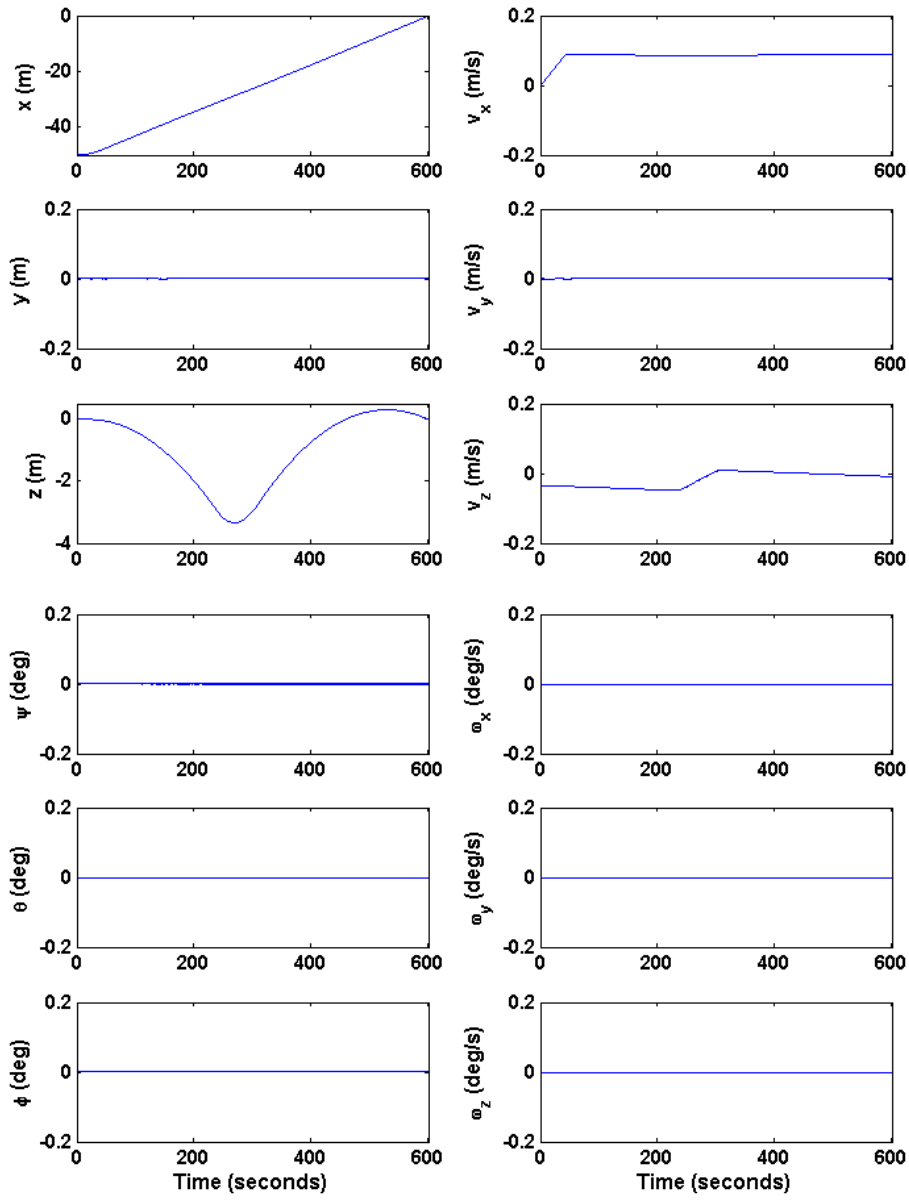

Fig. 30. Chaser Relative States, Test Case 3

being inactive most of the time. In summary, the variation in the chaser moments of inertia appeared to have little if any effect on the thrust profile of the vehicle. This is due to the relatively simple docking maneuver performed, which does not require excessive thrusting. If the thrusters were more active, the modeling errors would show greater effects on the system performance.

Figure 30 displays the behavior of the chaser vehicle's relative position, orientation, and rate states during the Test Case 3 example dock. Each of the graphs are similar to their counterparts from the example nominal and mass runs. The same z-position state drifting phenomenon occurs again while the vehicle proceeds smoothly towards the target in the positive x-direction, until the propulsion system corrects it back towards zero as it did in the previous cases. The orientation states are again well-behaved as before, never appearing to stray much from zero. Finally, all of the states converge as desired, so that the chaser successfully docks in 601.56 seconds with a final position error of less than 1.0 centimeter.

The VisNav sensor and integrated Kalman filter estimator performance results for the Test Case 3 example dock are presented in Figure 31. The performance is apparently identical to the nominal case, with the estimate errors for all of the states beginning far outside of the 3-σ error bounds; most of the states then converge at near range to well within the 3-σ bounds. As before, this is due to the range-dependency of the VisNav sensor calibration. However, since the controller was again able to effect docking without difficulty just as in the previous cases, thanks to the good convergence of the errors at close range, one can conclude that the VisNav sensor performance overall was satisfactory. Also, although the Kalman filter's velocity estimates again seemed overly conservative based on the width of their 3-σ bounds, just like the previous two Test Cases, the estimates themselves were accurate enough to help and not hinder the docking controller. Therefore, the integrated Kalman

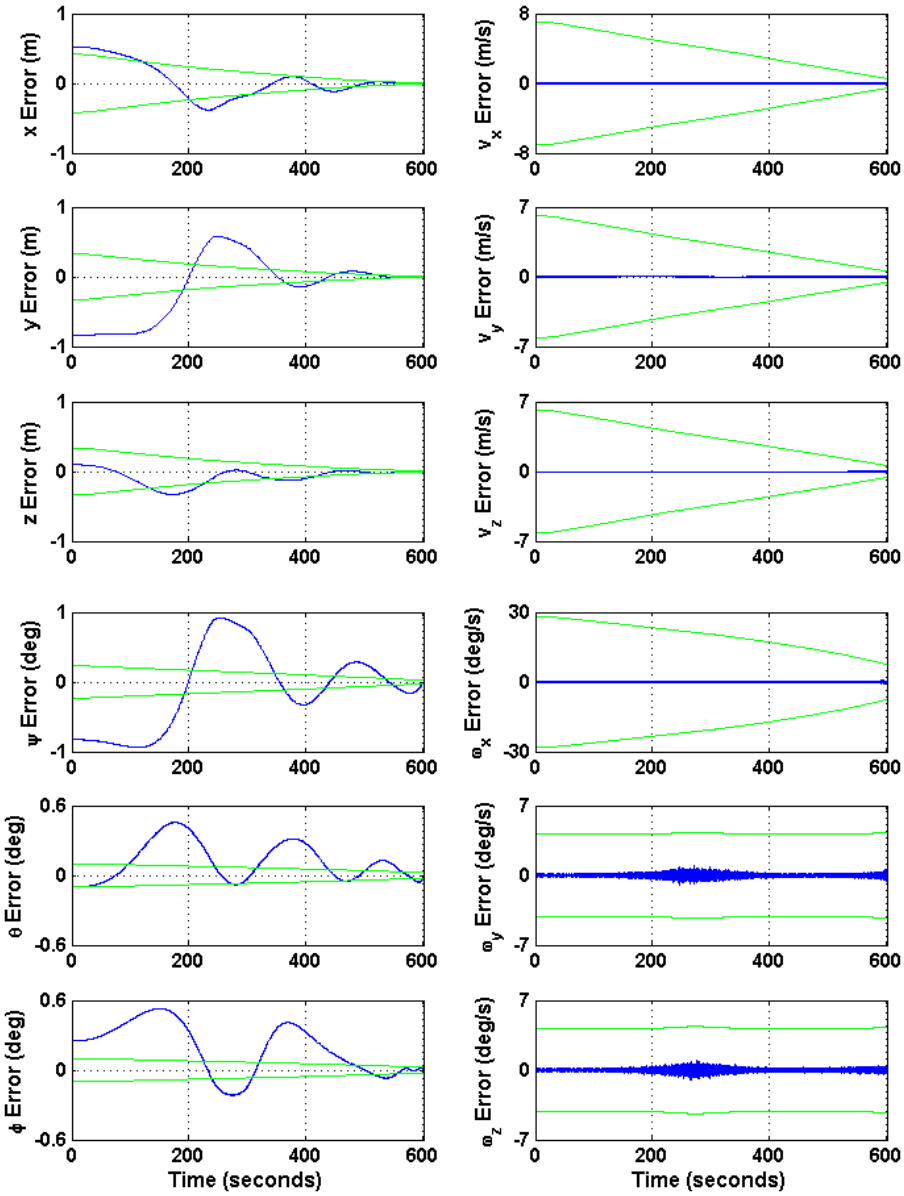

Fig. 31. VisNav & Kalman Filter Estimation Error, Test Case 3

filter's performance in the Case 3 tests was deemed satisfactory.

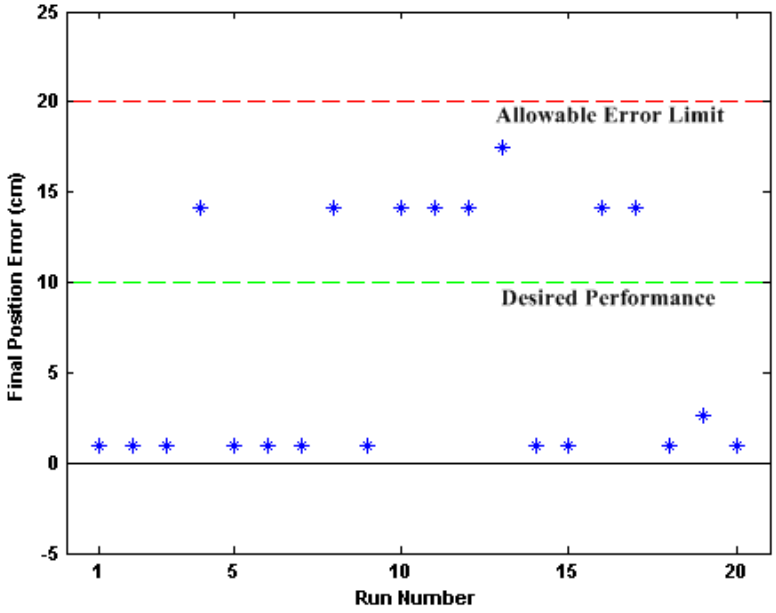

Fig. 32. Final Position Error Results Summary, Test Case 3

Test Case 3 consisted of twenty simulation runs as specified in the experimental test plan from Chapter VII. The purpose of this set of tests was to evaluate the docking controller's performance robustness with respect to uncertainty in modeling the chaser moments of inertia. Test Case 3 results are summarized in Figures 32, 33, and 34. These figures illustrate how each of the runs performed with respect to the design criteria in final position error, final time error, and final total velocity respectively. By comparing the graphs to those from the nominal case, it is apparent that moment of inertia modeling uncertainty does not significantly affect the controller's performance for the specified docking scenario. This is a somewhat surprising result that might not fully generalize to more complicated docking maneuvers or starting position arrangements; however, for the scenario of present interest this discovery is strongly positive. Sub-section 5 will contain further analysis of the Test Case 3 results

in context with the rest of the controller #1 portion of the experiment.

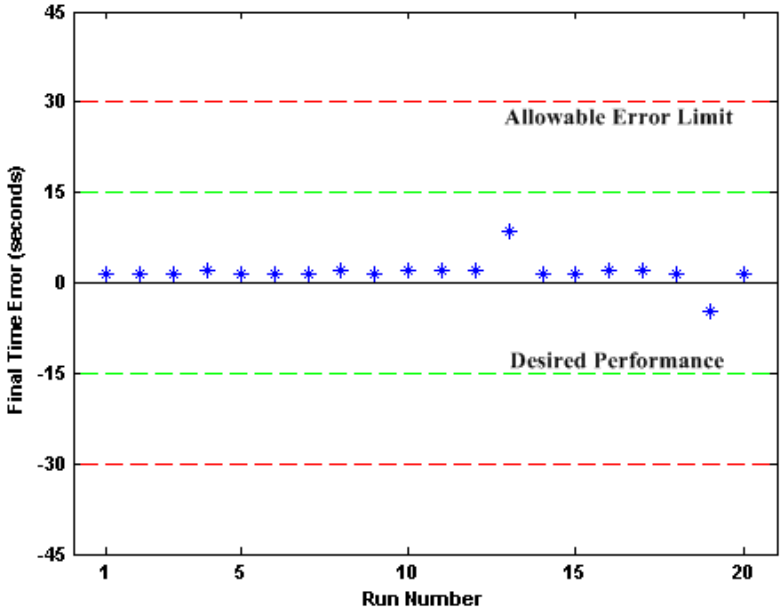

Fig. 33. Final Time Error Results Summary, Test Case 3

4. Test Case 4: Starting Position Uncertainty

Test Case 4, the final case for controller #1, evaluated the performance robustness of the docking controller with respect to maneuver starting position uncertainty. The scenario used was otherwise the same as the nominal case; there was also sensor noise present during the test runs. As specified in the Chapter VII experimental test plan, the chaser vehicle's relative starting position was randomly selected for each run using a Gaussian pseudo-random number generator. The output of the generator was then scaled so that its mean was the nominal starting position value of 50 meters. The generator's standard deviation was adjusted to be 5% of the nominal position, or 2.5 meters. All other scenario parameters such as chaser vehicle mass and chaser vehicle moments of inertia were held constant at their nominal values for the duration of all

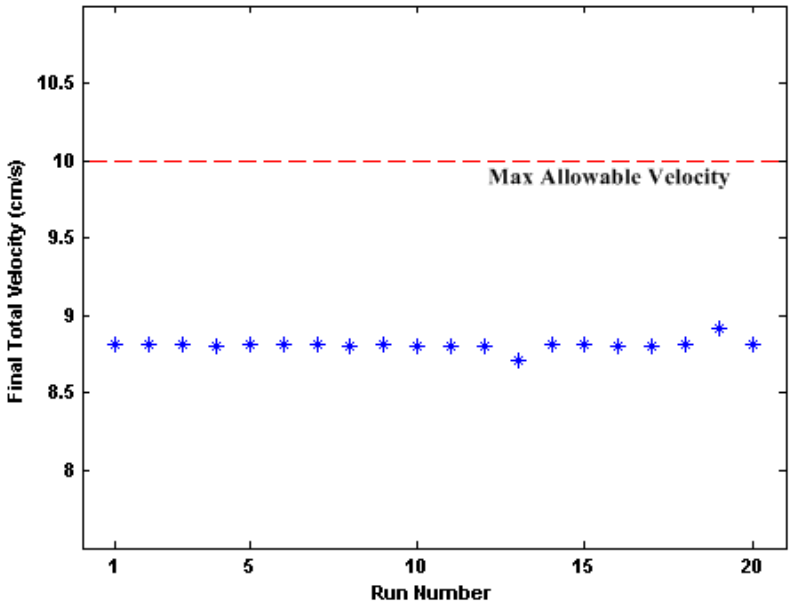

Fig. 34. Final Total Velocity Results Summary, Test Case 3

Test Case 4 runs.

The relative trajectory of an example Test Case 4 simulation run is shown in Figure 35. As it was for all other cases, the trajectory is expressed here in target frame coordinates. However, since relative starting position was the parameter being evaluated in this Test Case, the maneuver did not begin at a fixed position for each run as for the other tests. For the example shown, the starting position was at approximately $(-49.90, 0, 0)$ meters relative to the target vehicle, and concluded in a successful dock when contact was made with the target vehicle in the near neighborhood of $(0, 0, 0)$ meters.

Figure 36 presents the time history of the chaser vehicle's total relative velocity magnitude for the Test Case 4 docking example. There are clear similarities between this graph and the total velocity plots from the previous cases. For example, the total relative velocity was less than the maximum allowable final velocity throughout

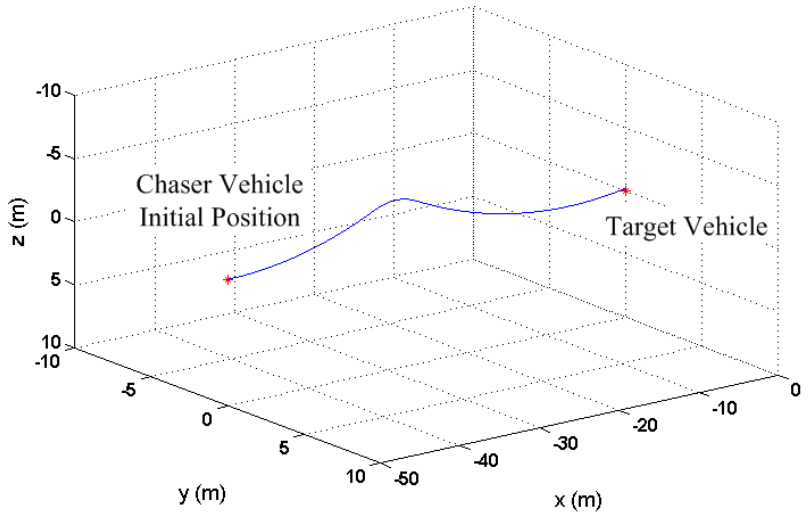

Fig. 35. Docking Relative Trajectory, Test Case 4

the maneuver, just as it was for each of the previous plots. Docking between the vehicles also occurred at a similar relative velocity as for the other cases—here, the total relative contact velocity was approximately 8.6 centimeters per second, which was slightly slower than in the previous examples. Thus, the end result for Case 4 was also the same as for the other cases, in that it met the total relative velocity criteria for a successful dock.

Figure 37 illustrates the chaser vehicle's thruster activity during the example Test Case 4 maneuver. The thrust profile is seen to be qualitatively similar to the thrust profiles of the previous cases. There was a similar period of heavy x-axis thruster activity in the initial stages of the maneuver, followed by a coasting period. Then, the z-axis thrusters briefly fired to correct vehicle drift in that axis. Finally, the vehicle returned to coasting until successfully docking with the target vehicle at maneuver completion. Notice that just as for the other cases, the y-axis thrusters were never used during the Case 4 example maneuver. Thus, the qualitative appearance of the

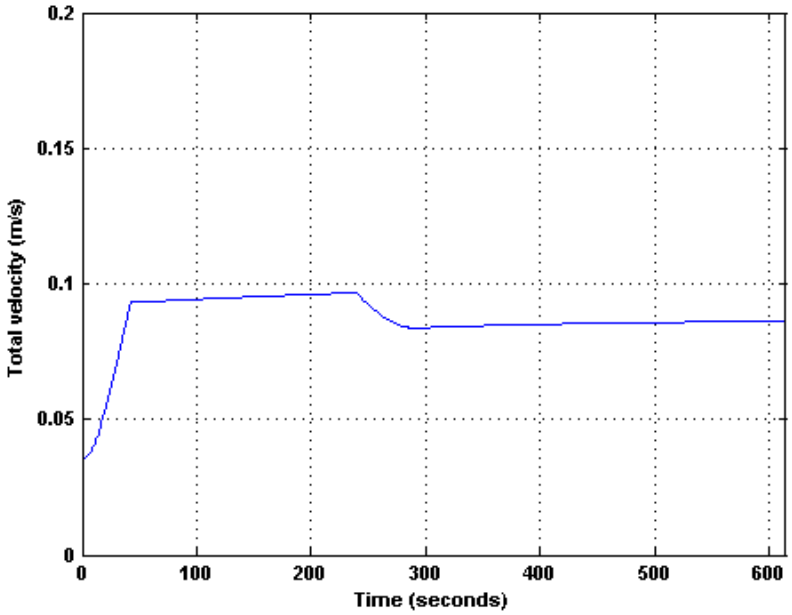

Fig. 36. Total Relative Velocity, Test Case 4

chaser vehicle's thrust profile shows little effect due to variations in the parameter investigated in this Test Case.

Time histories of the chaser vehicle's relative position, orientation, and rate states are displayed in Figure 38 for the Test Case 4 example dock. These graphs are again similar in general appearance to those from the previous cases: the vehicle drifts towards negative $z$in position while proceeding smoothly towards the target in the positive x-direction. Eventually, the z-axis drift is corrected by the propulsion system once it reaches a triggering threshold value, and the orientation states again never appear to stray much from the near zero region. However, even though this run appears similar to the other examples and was a successful dock, it was not as desirable of a docking attempt. The final chaser position error was approximately 19.8 centimeters, and docking did not occur until 614.31 seconds of simulation time had elapsed; both of these conditions clearly show a greater amount of error than the

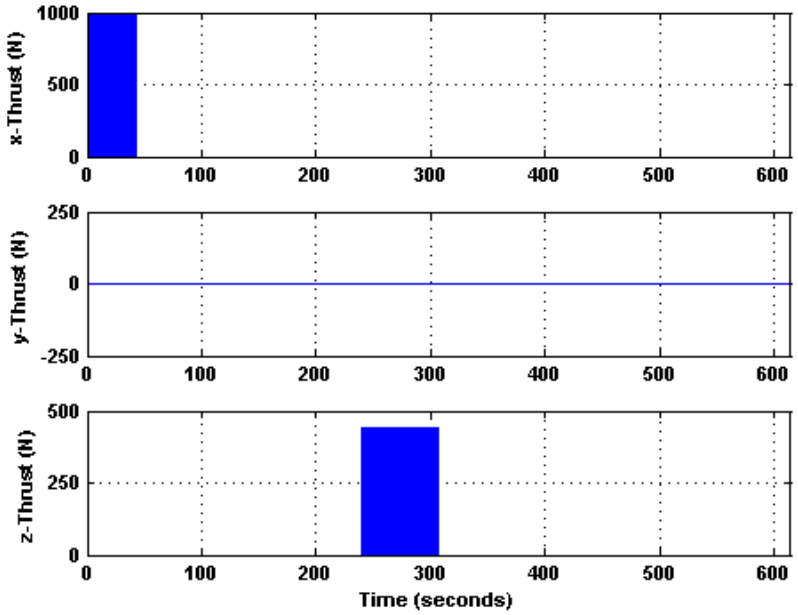

Fig. 37. Chaser Thrust Profile, Test Case 4

other test cases did.

Figure 39 shows the VisNav sensor and integrated Kalman filter performance results for the Test Case 4 example maneuver. The performance is apparently identical to the previous cases, with the initial estimate errors for all states lying far outside the 3-σ error bounds. Most of the states then converge at near range to well within the 3-σ bounds. As before, this is due to the range-dependent VisNav sensor calibration method currently employed. However, the errors exhibited identical close range convergence characteristics to the previous examples. Thus, one can conclude that the VisNav sensor performance for this case was satisfactory overall. By inspecting the velocity graphs, it seems that the Kalman filter's velocity estimates were again overly conservative. However, the estimates themselves were just as accurate as for the other cases. Therefore, the integrated Kalman filter's performance in the Test Case 4 simulation runs was deemed satisfactory as well.

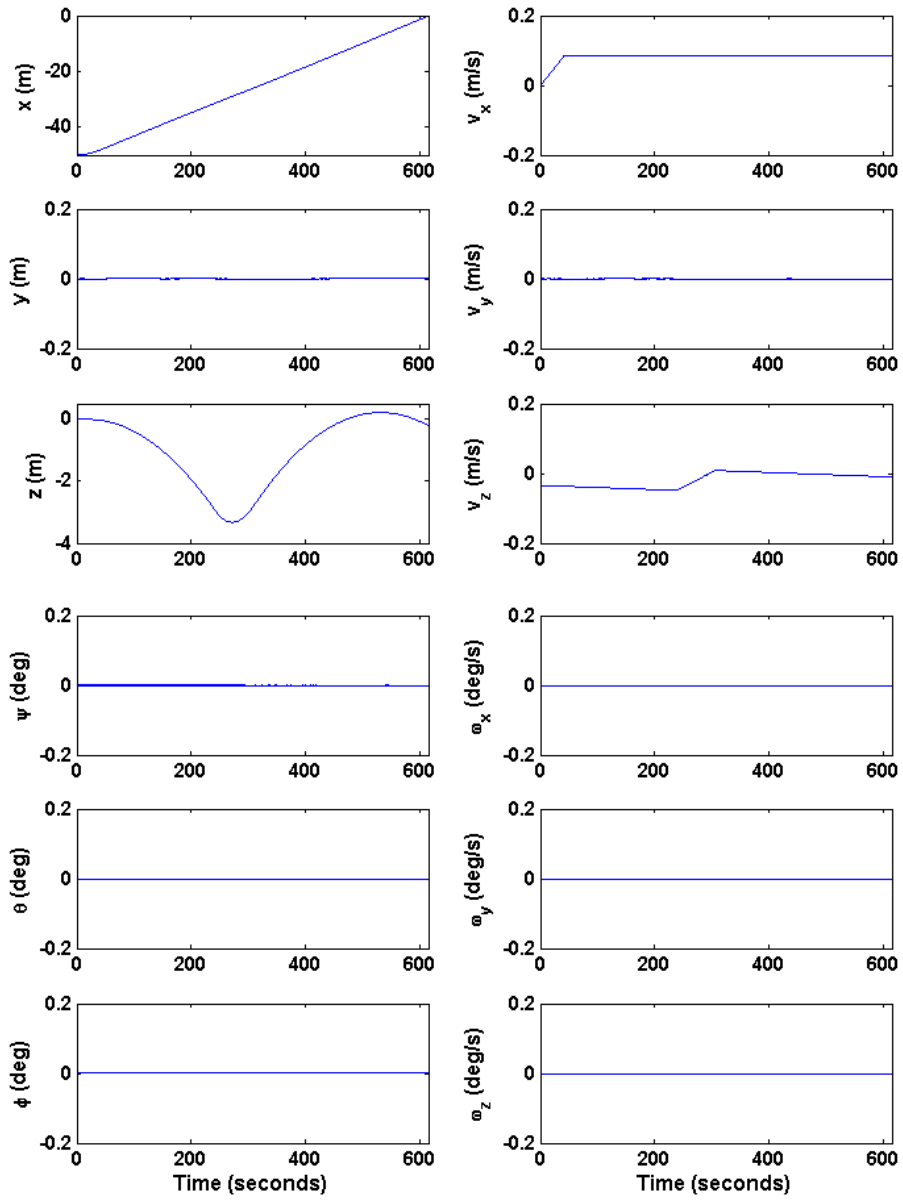

Fig. 38. Chaser Relative States, Test Case 4

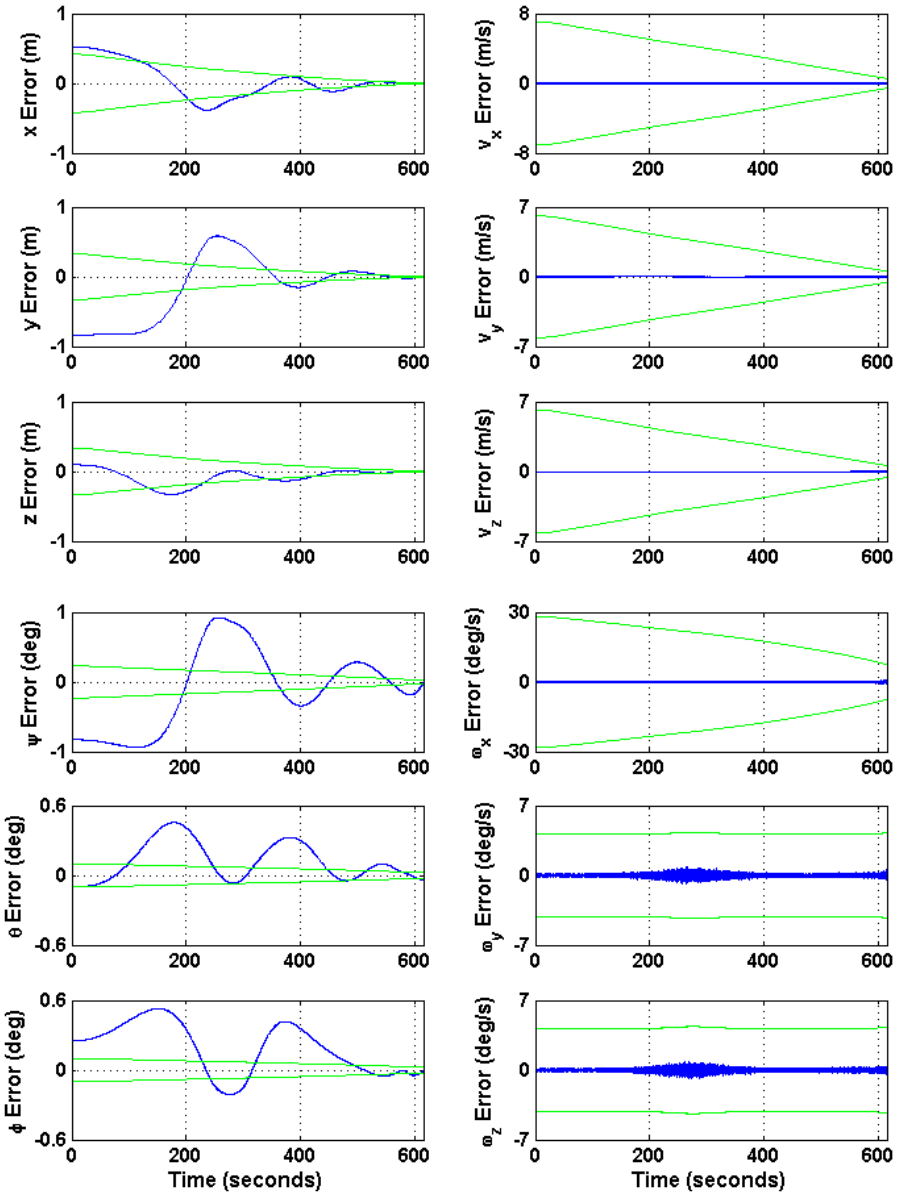

Fig. 39. VisNav & Kalman Filter Estimation Error, Test Case 4

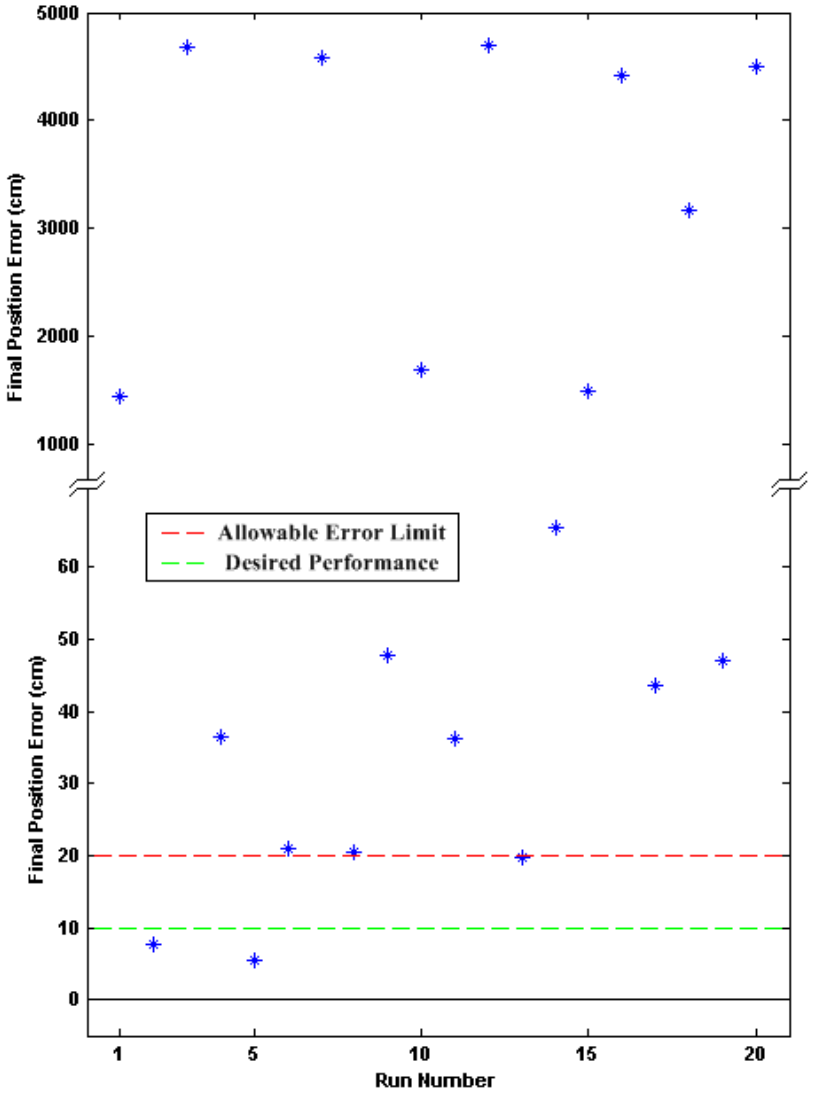

Fig. 40. Final Position Error Results Summary, Test Case 4

Twenty simulation runs were performed to fulfill the Test Case 4 test plan specified in Chapter VII. The purpose of these tests was to evaluate the docking controller's performance robustness with respect to relative starting position uncertainty. Figures 40, 41, and 42 illustrate how each of the runs performed with respect to the design criteria in final position error, final time error, and final total velocity respectively. As displayed in the figures, the controller's performance is very start position dependent: only one attempt was successful, and even relatively small percentage variations from the nominal led to complete failure to meet docking success criteria. This is due to the tuning of the controller gains. The controller as designed is very fuel efficient because the gains are very small, resulting in low thruster activity levels. However, while this results in excellent performance in nominal conditions, the controller has a reduced ability to reject large disturbances such as those resulting from starting position errors. Thus, it must be concluded that the controller as tuned is non-robust to initial starting position uncertainty. The next sub-section contains further analysis of the Test Case 4 results in the context of the rest of the controller #1 portion of the experiment.

5. Summary of Results

This section presents the full results of the experimental test plan used to evaluate automated spacecraft docking controller #1. Successful docking was demonstrated at least once for each of the test cases run, each of which evaluated the controller's performance robustness to one parameter in addition to sensor noise. The VisNav sensor and integrated Kalman were found to perform satisfactorily for all test cases, especially in the near range region. It does not appear that their performance made a significant impact upon whether or not the controller was able to dock, since the estimate errors were very similar for all runs whether docking occurred or not.

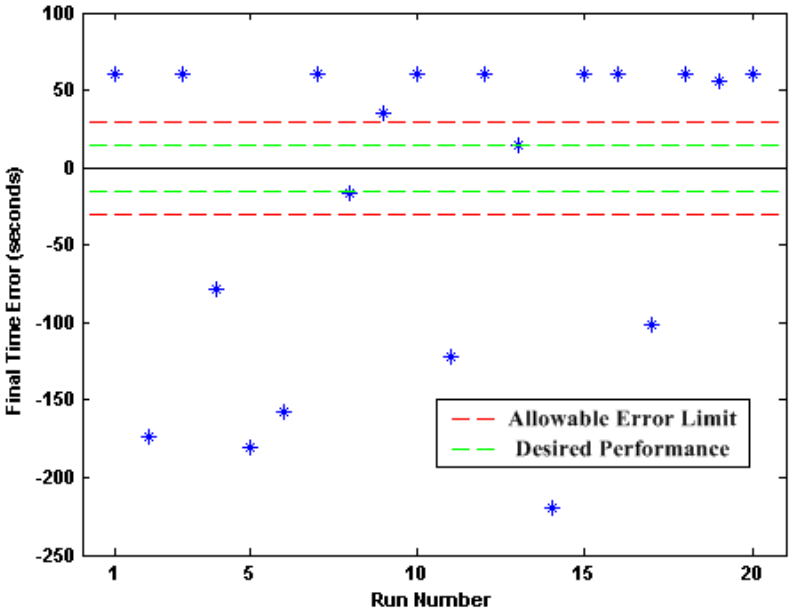

Fig. 41. Final Time Error Results Summary, Test Case 4

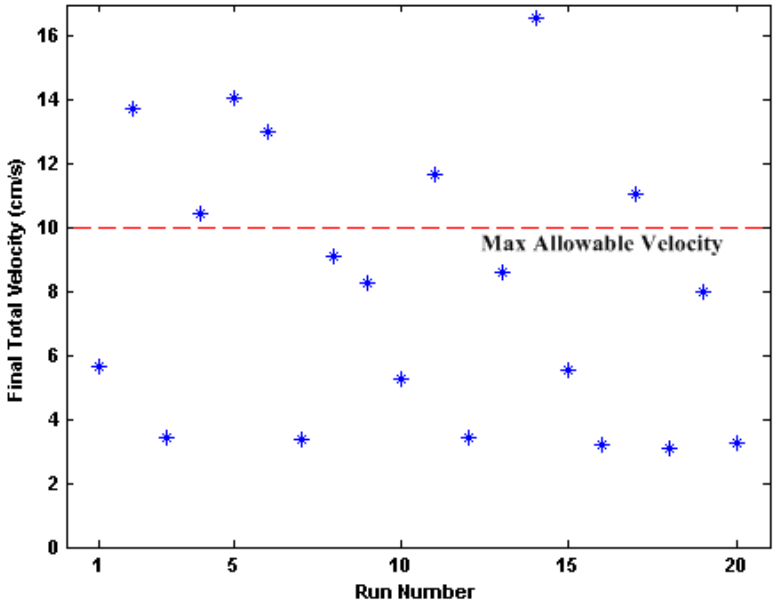

Fig. 42. Final Total Velocity Results Summary, Test Case 4

The experimental results from the different test cases can be generally summarized as follows:

- In nominal conditions, consisting of sensor noise only, the controller was successful on every attempt. Nearly half of the time it achieved the more stringent 'preferred' docking standards.

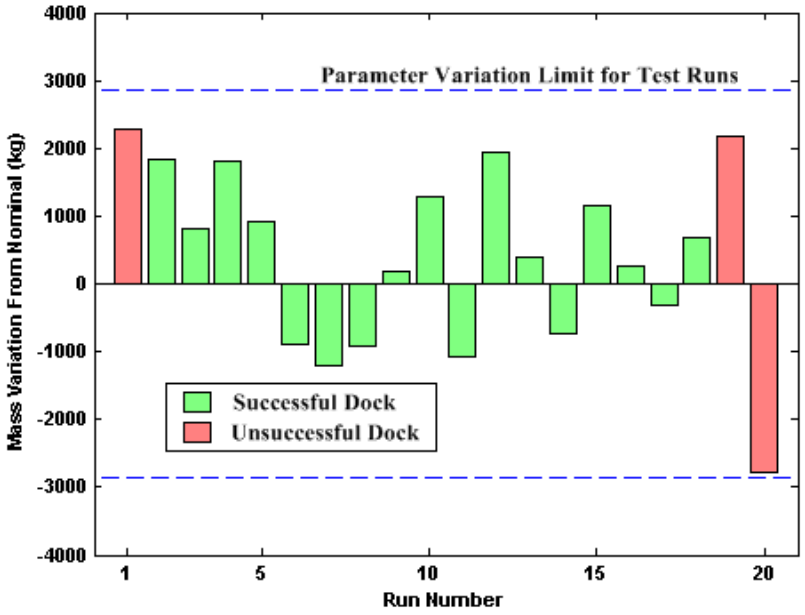

Fig. 43. Parameter Variation & Docking Result Summary, Test Case 2

- Chaser mass modeling uncertainty did affect docking success somewhat. As shown in Figure 43, the greater the modeling error, the greater the likelihood of failure. However, overall docking success for this case was still 85%, and the only failed attempts were for mass errors of greater than ±10.5%, or ±2000 kilograms. Thus, one can conclude that within a reasonably large error region, the docking controller is robust to mass modeling uncertainty.

- Chaser moment of inertia modeling uncertainty did not affect docking success

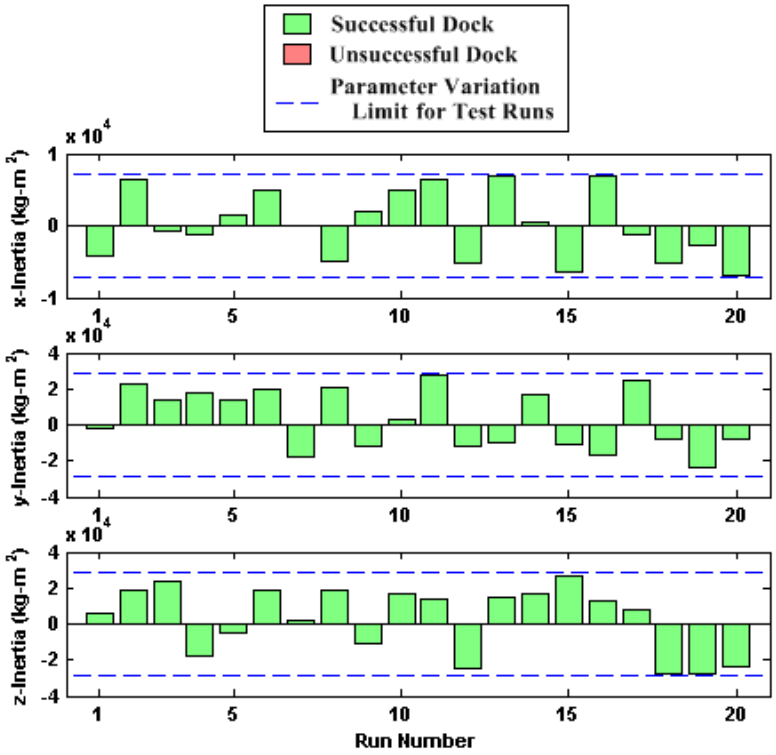

Fig. 44. Parameter Variation & Docking Result Summary, Test Case 3

for the specified scenario. Figure 44 shows that the controller was successful for every attempt, even though the randomly chosen inertia values covered the entire test range of ±15% of nominal. Though this result might not fully generalize to more complicated docking maneuvers or starting conditions, it demonstrates that the VisNav-Controller system is a promising candidate for the scenario for which it was designed.

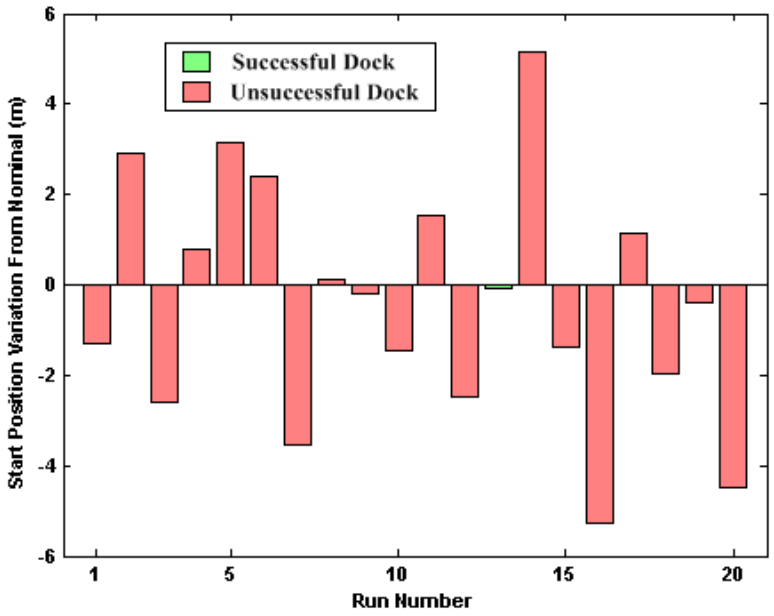

Fig. 45. Parameter Variation & Docking Result Summary, Test Case 4

- The controller as tuned is not robust to initial starting position uncertainty. Only one docking attempt was successful, and Figure 45 illustrates the controller's inability to compensate for even relatively small percentage errors in initial position. The controller gains are simply too small to adequately compensate for starting position uncertainty. The controller should be re-tuned for better performance robustness with respect to initial starting position errors.

The test case simulation run results are further summarized in Table V. The

mean and standard deviation for each test case is presented to illustrate the controller's precision and accuracy for the various test cases. Notice that the means and standard deviations for the different final errors are very tightly grouped for all of the cases except for position variation. This reinforces the notion of the controller's sensitivity to this parameter, since relatively small changes in initial position led to much more widely varied final conditions than for the other cases.

Table V. Docking Test Cases Results Summary, Controller #1

Test Case	Nominal Case	Mass Var.	Inertia Var.	Position Var.
Runs	20	20	20	20
Successful Docks	20	17	20	1
Position Error (centimeters)	$\mu = 8.82$ $\sigma = 6.616$	$\mu = 8.42$ $\sigma = 5.973$	$\mu = 6.50$ $\sigma = 6.820$	$\mu = 1551.6$ $\sigma = 1971.5$
Time Error (seconds)	$\mu = 1.51$ $\sigma = 2.627$	$\mu = 5.22$ $\sigma = 19.50$	$\mu = 1.75$ $\sigma = 2.193$	$\mu = -20.25$ $\sigma = 101.6$
Final Velocity (cm/s)	$\mu = 8.81$ $\sigma = 0.040$	$\mu = 8.77$ $\sigma = 0.276$	$\mu = 8.81$ $\sigma = 0.033$	$\mu = 8.04$ $\sigma = 4.300$

In order to determine over what range of starting position values the controller *could* successfully dock, additional start position variation tests were conducted after the test plan was concluded. Through a series of test runs in which the starting position was manually varied, it was determined that the controller could successfully dock for starting positions of approximately 49.90 meters to about 50.25 meters, or -0.2% to $+0.5\%$ of nominal. This is obviously much too small of an operational range for practical applications, so further work is required to improve controller #1's robustness to starting position uncertainty.

B. Results for Controller #2

1. Nominal Conditions

Just as it was for controller #1, the nominal case for controller #2 is an automated spacecraft docking maneuver performed with the only disturbance being sensor noise. (Recall that this noise is the result of the VisNav sensor model and the Kalman filter rate estimator that are part of the docking system.) As before, all chaser vehicle properties such as mass and moments of inertia, as well as chaser relative starting position and relative initial attitude, retain their nominal values for the duration of the nominal simulation run.

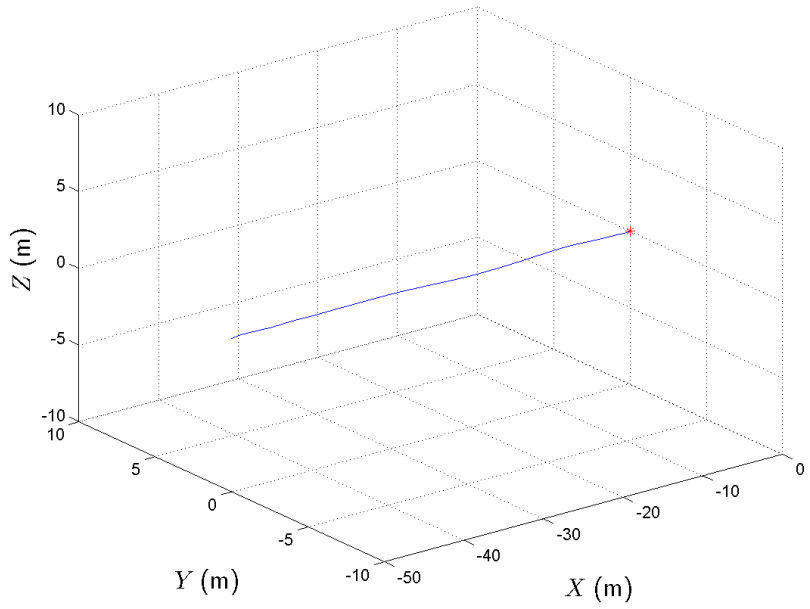

Fig. 46. Chaser Relative Trajectory, Nominal Case

The chaser's relative trajectory in target frame coordinates for the nominal case is shown in Figure 46. As it should be, the chaser vehicle is initially at position $(-50, 0, 0)$ meters in this coordinate system, and it steadily closes the distance to

the docking point at $(0, 0, 0)$ meters. The nominal trajectory is much smoother when using controller #2 than it was with controller #1.

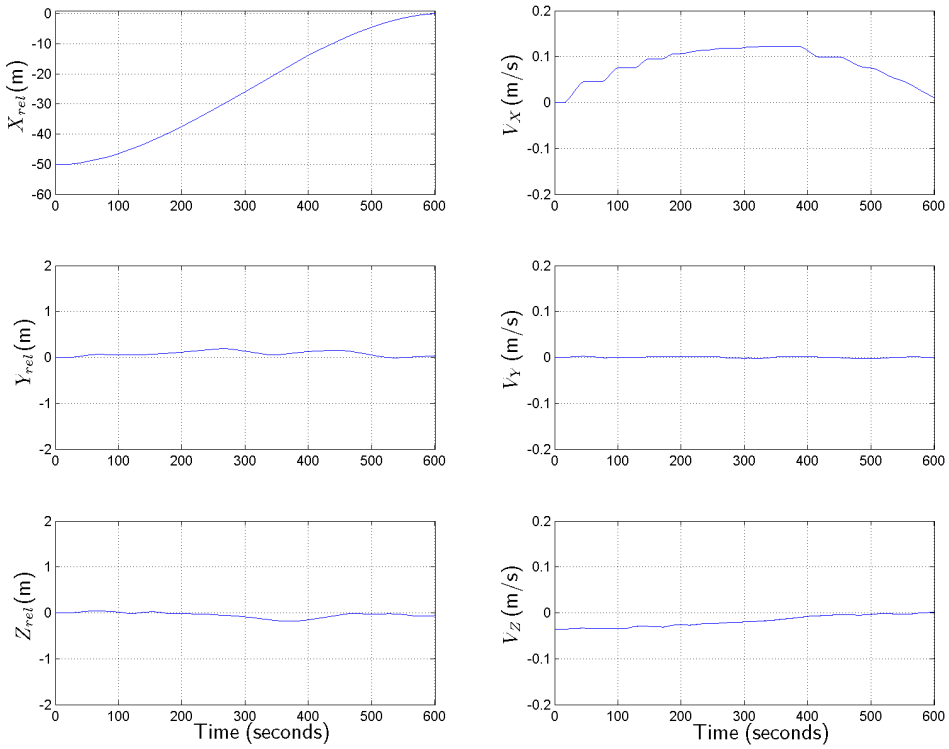

Fig. 47. Chaser Relative Position & Velocity, Nominal Case

The chaser vehicle's relative position, orientation, and rate states during the controller #2 nominal maneuver are shown in Figures 47 and 48. Note the smooth behavior of the vehicle in the along-track (x) direction, which follows very closely the proscribed spline curve it was commanded to follow. Position components y and z and their derivatives are also nicely constrained during the maneuver. The 3-2-1 Euler orientation angles of the vehicle smoothly converge as commanded from their given $\pm 3°$ initial conditions, also closely following the desired "paths" commanded by the controller. The angular velocity states show the effect of the torques exerted

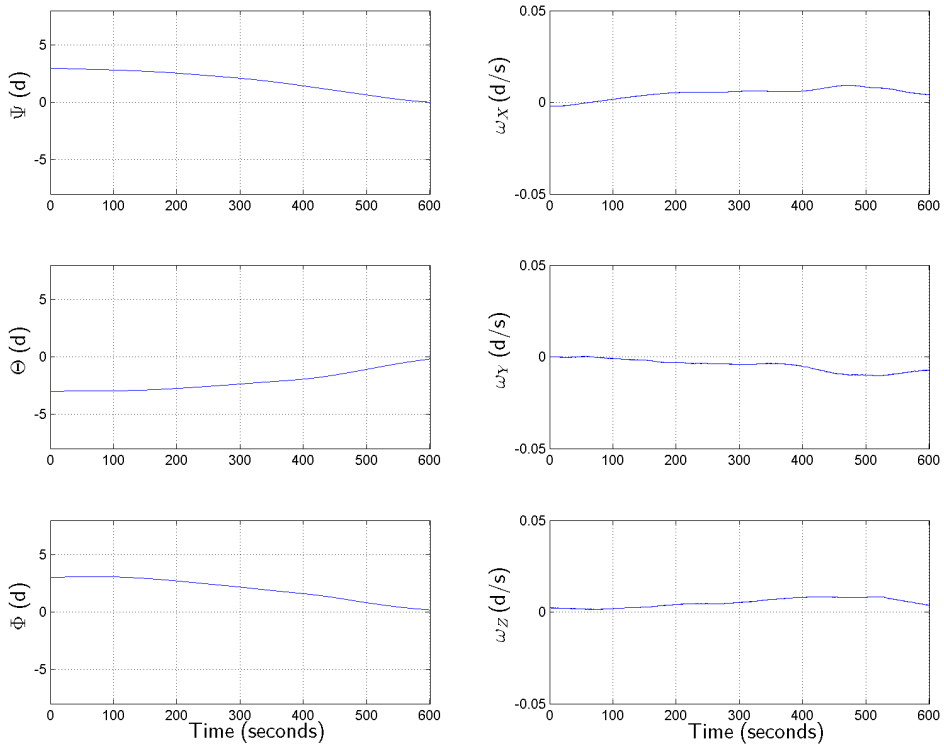

Fig. 48. Chaser Relative Orientation & Attitude Rate, Nominal Case

by the reaction wheels to bring the attitude to zero, but remain satisfactorily small throughout in spite of that. The final total position error of 6.71 cm is well below the requirement for a successful dock, as are the final attitude errors of 0.01, -0.19, and 0.19 degrees respectively in yaw, pitch, and roll. Finally, the docking was completed in 600.29 seconds of elapsed simulation time, also easily exceeding the time requirement for success.

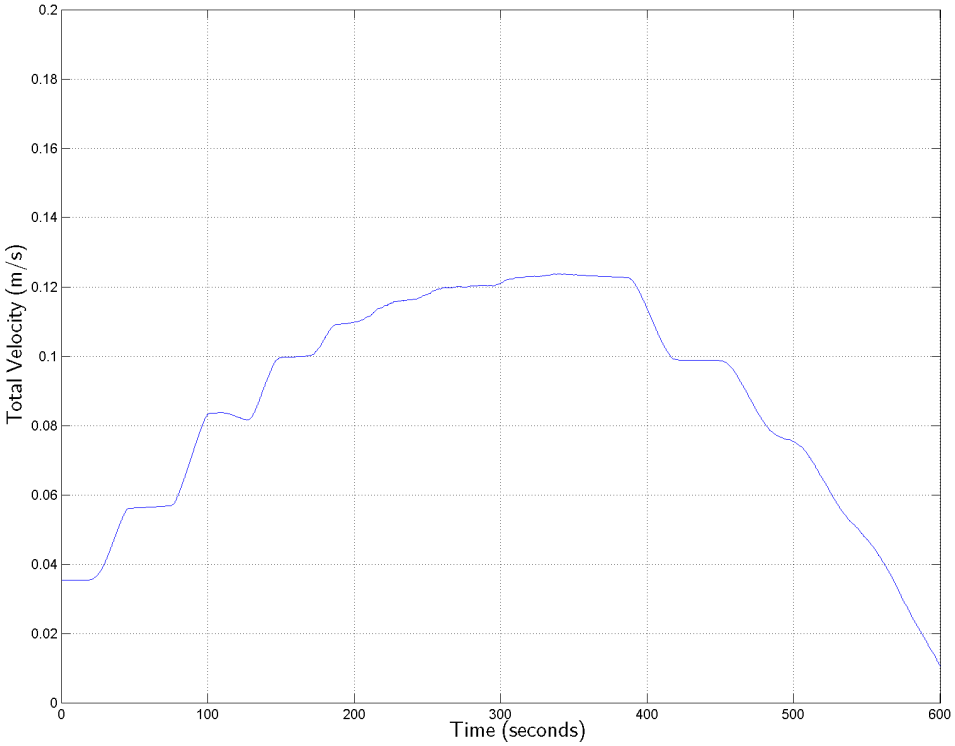

Fig. 49. Relative Velocity Magnitude Profile, Nominal Case

Figure 49 shows the time history of the chaser vehicle's total relative velocity magnitude during the nominal maneuver. It is evident that the total velocity magnitude was well-constrained throughout the maneuver, ensuring that the vehicle proceeded at a suitably slow velocity at all times. The velocity magnitude at docking

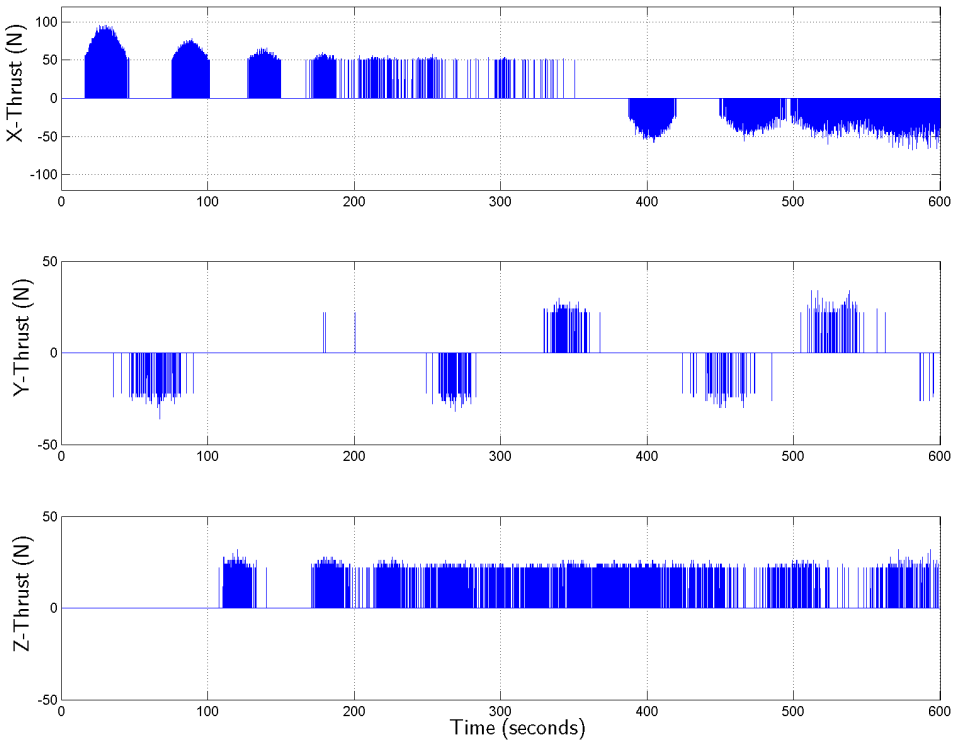

Fig. 50. Chaser Thrust Profile, Nominal Case

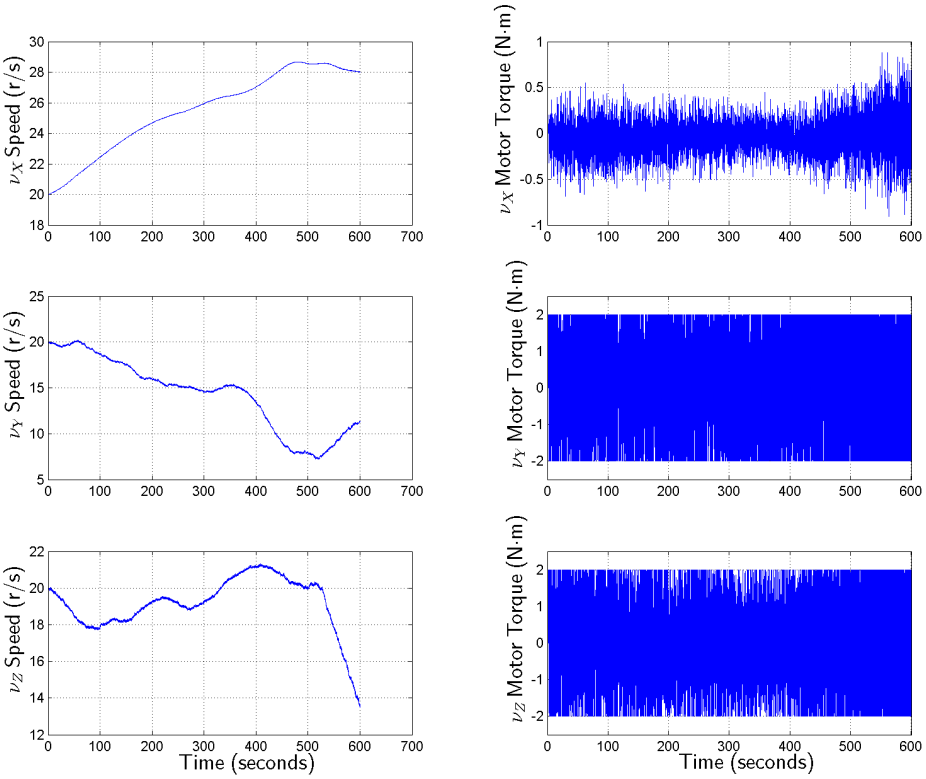

Fig. 51. Wheel Speeds & Motor Torques, Nominal Case

is approximately 1.06 centimeters per second, verifying that the relative velocity at contact between the vehicles easily was well slower than the desired velocity limit of 10 centimeters per second. It also easily cleared the "preferred" criteria of 5 cm/s, as did the attitude angles previously discussed; thus, the nominal case is considered an "exceptional" run (for passing all preferred criteria).

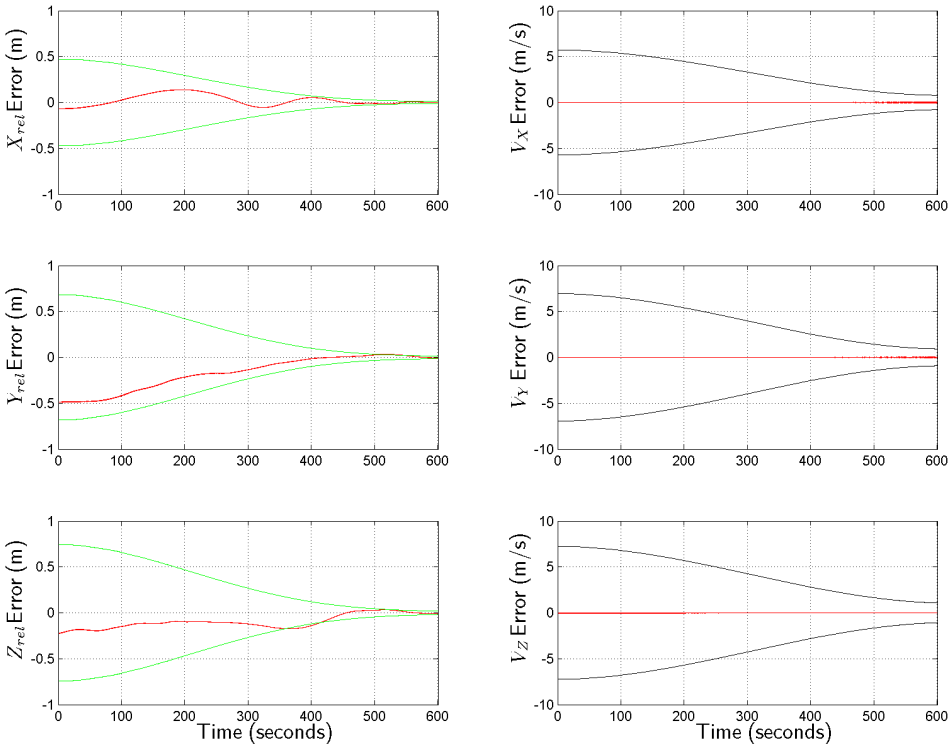

Fig. 52. Position & Velocity Estimate Errors, Nominal Case

The chaser vehicle's applied thrust history for the nominal case is presented in Figure 50. Consistent with the velocity and position plots already discussed, most of the work is done by the x-axis thrusters to follow the spline trajectory to the docking point. The role of the y and z thrusters is largely one of maintenance or cleanup, since those position components began at zero. However, thanks to the in-orbit-plane

coupling of x and z, the z thrusters do exhibit a bit more activity in the middle and end of the run to counteract the motion "bleedover" from x.

The reaction wheel angular velocities and control motor torques are plotted in Figure 51. The y- and z-wheel motors spend much of the run at either one of their allowable torque limits or the other, exhibiting more chatter than would be ideal. However, the net motion of the wheels is reasonable and the behavior of the attitude angles is excellent as previously shown. Thus, the overall attitude control system behavior is definitely acceptable.

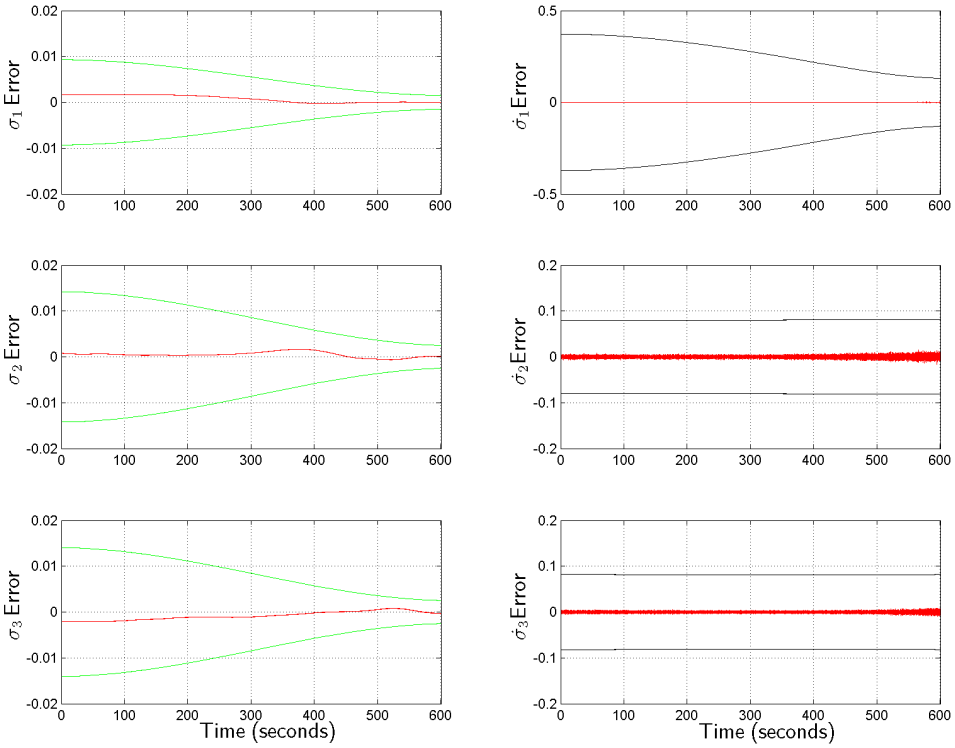

Fig. 53. Orientation & Attitude Rate Estimate Errors, Nominal Case

Figures 52 and 53 show the estimation accuracy of the VisNav sensor (for position and orientation) and the integrated Kalman filter (for velocity and attitude rate)

during the nominal docking run. Note that while position and velocity are shown terms of meters as before, the orientation and orientation rate are now shown in terms of Modified Rodrigues Parameters (MRPs). Since both the sensor and the Kalman filter report in MRPs and their derivatives for both their estimates and uncertainty bounds, it seemed sensible to show the information directly rather than possibly lose something while converting to a different parameterization. Note that all estimates are well-behaved, basically lying properly within their 3-σ error bounds for at least $\sim 99.7\%$ of the time as they should. Also, all of the component uncertainty bounds that are supposed to converge do so quite well; the bounds for the second and third components of the MRP derivative also act as they should by basically remaining steady-state for the duration of the run. In short, the performance of both VisNav and the Kalman filter was quite satisfactory througout the maneuver. This fact certainly contributed to the excellent results obtained from controller #2 in this case.

The results for this nominal case will be further analyzed in the context of the complete controller #2 experimental results in Sub-section 14.

2. Test Case 1: Max Positive Position Error

Test Case 1 for controller #2 evaluated the performance robustness of the docking controller when given a maximum *positive* starting position uncertainty. Thus, initially the chaser was positioned at 57.5 meters behind the docking point, which corresponds to a 3-σ position value from the controller #1 test plan. All other conditions for the test run were the same as the nominal case, such as chaser mass, chaser moments of inertia, and initial relative attitude; there was also sensor noise from VisNav and the Kalman filter just as in the nominal.

The relative trajectory in target frame coordinates for the "max position" test

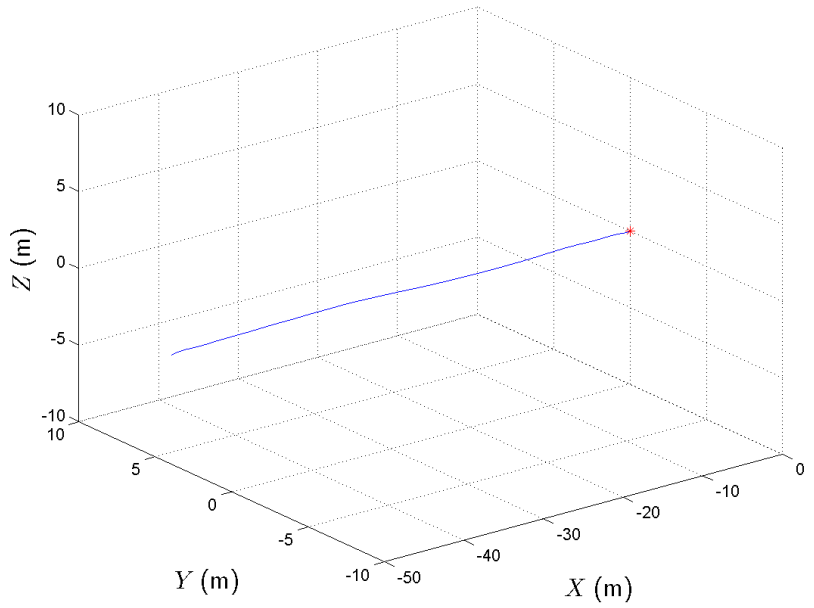

Fig. 54. Chaser Relative Trajectory, Max Position Case

case is shown in Figure 54. Note that its behavior looks very similar to the nominal; the controller seems to have accomodated the maximum 3-σ position error quite well. Interestingly, this is in stark contrast to the position results from controller #1, which simply could not dock successfully with such a large position error.

The relative states for Test Case 1 are shown in Figures 55 and 56. Note that they resemble the nominal results very closely, other than the obvious difference in initial x-axis position. Aside from that, the characteristics of each plot are very similar to their nominal counterparts. The final total position error for Test Case 1 is 6.13 cm, while the final attitude errors are 0.01, -0.31, and 0.18 degrees respectively in yaw, pitch, and roll—all of which easily eclipsed the required values for a successful dock. Finally, the docking was completed in 602.25 seconds of elapsed simulation time, also easily exceeding the time requirement for success.

Figure 57 shows the total relative velocity magnitude time history for the first

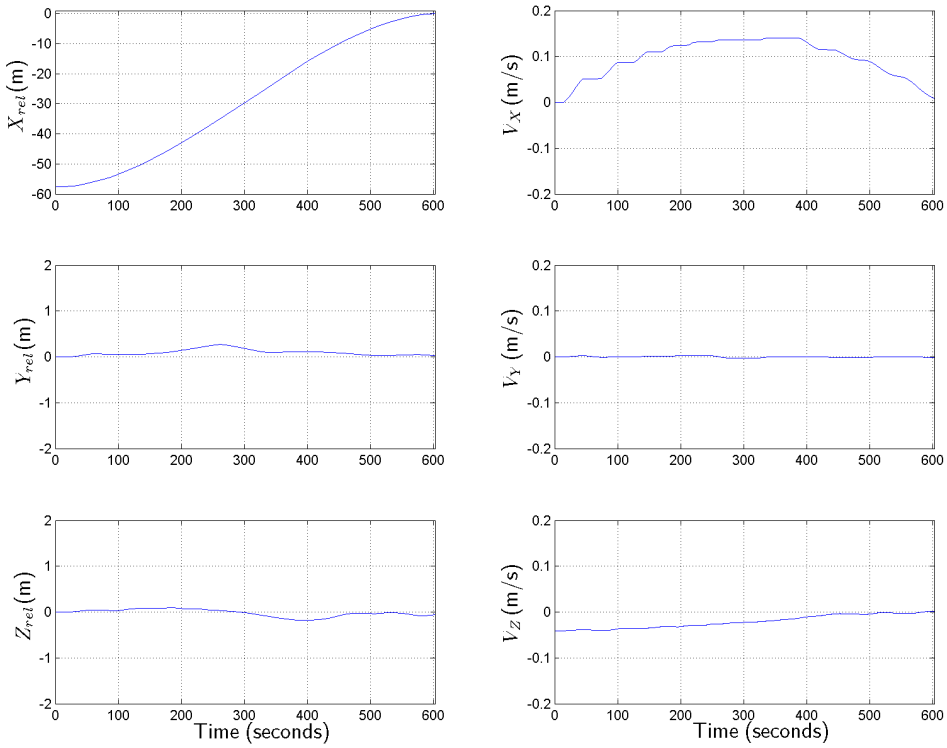

Fig. 55. Chaser Relative Position & Velocity, Max Position Case

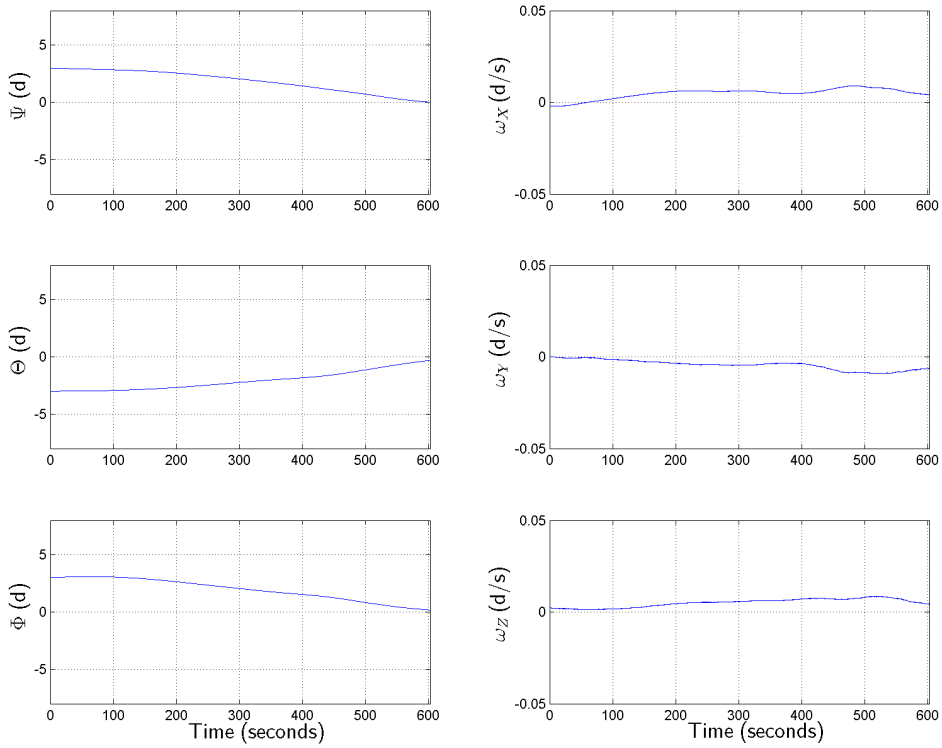

Fig. 56. Chaser Relative Orientation & Attitude Rate, Max Position Case

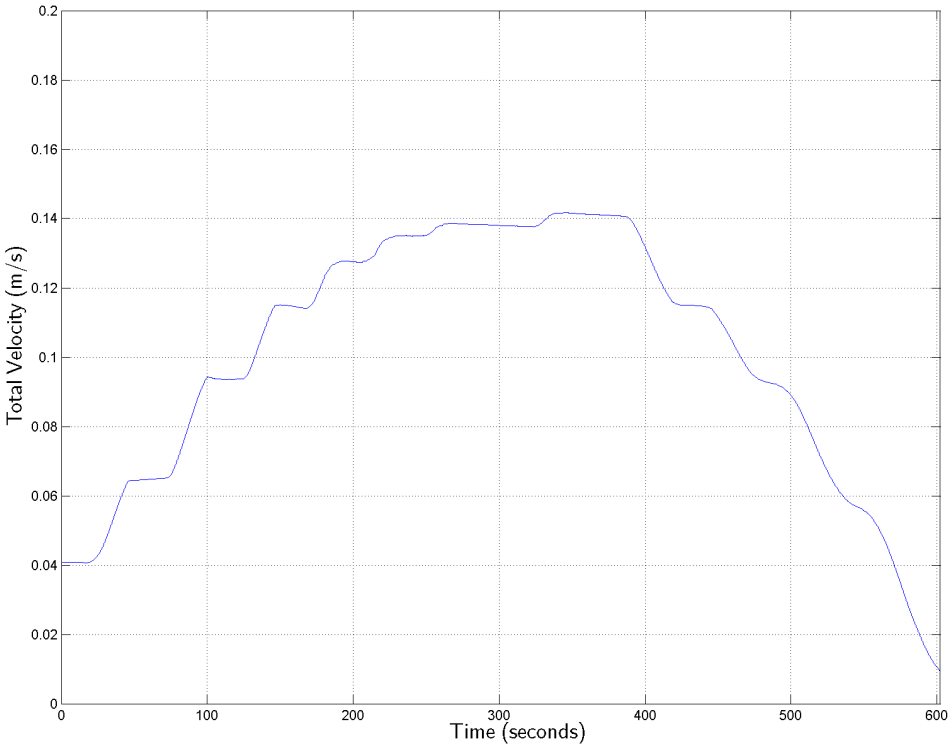

Fig. 57. Relative Velocity Magnitude Profile, Max Position Case

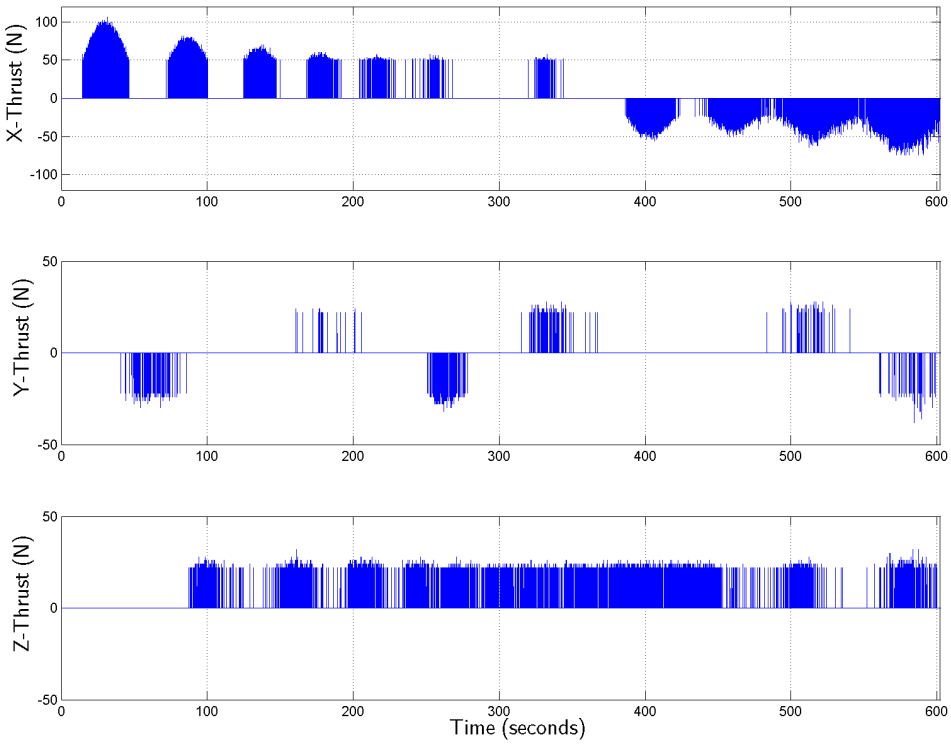

Fig. 58. Chaser Thrust Profile, Max Position Case

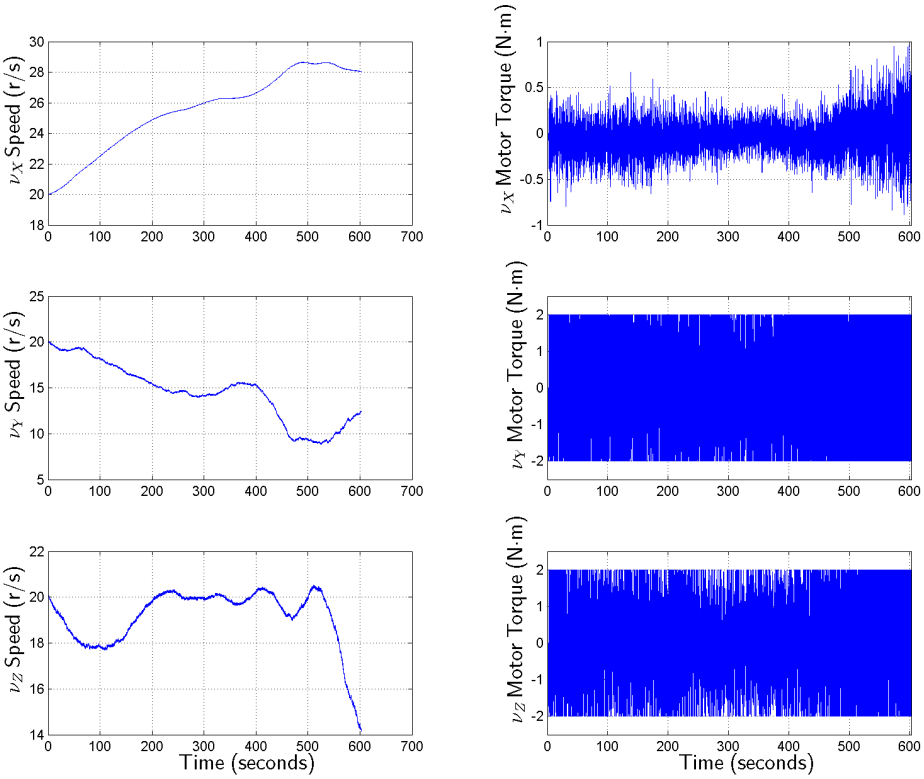

Fig. 59. Wheel Speeds & Motor Torques, Max Position Case

test case. While the value has slightly more variation in it than the nominal one, it still shows very desirable behavior particularly near the end. The velocity magnitude at docking is approximately 0.93 centimeters per second, which is actually slower than the nominal case docked and well below both the desired velocity limit (10 cm/s) and the preferred limit (5 cm/s). Based on this and the final relative attitude at dock, Test Case 1 is also considered an "exceptional" run, just as the nominal was.

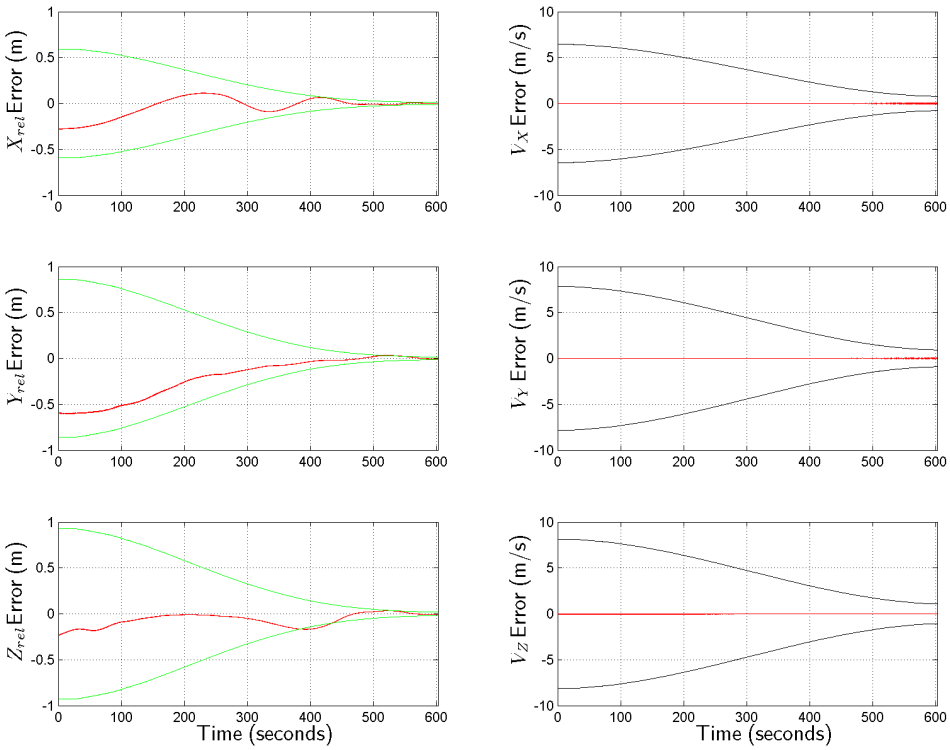

Fig. 60. Position & Velocity Estimate Errors, Max Position Case

The rest of the results for the "max position" test case for controller #2 are presented in Figures 58 through 61. The graphs are so similar to the plots for the nominal case that they will be allowed to speak for themselves. They show, in short, that the controller easily handled the positive 3-σ position error case, with results

very similar to the nominal; the VisNav sensor and the Kalman filter also performed satisfactorily as they did in nominal conditions.

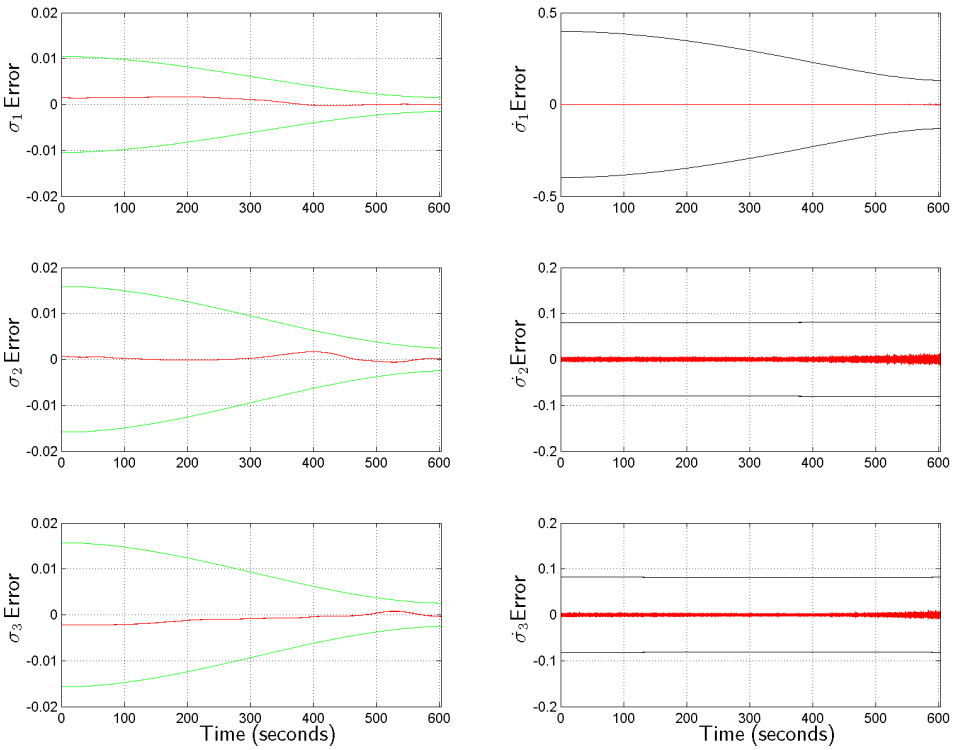

Fig. 61. Orientation & Attitude Rate Estimate Errors, Max Position Case

These "max position" case results will be briefly discussed in the context of the full controller #2 experiment in Sub-section 14.

3. Test Case 2: Max Negative Position Error

The second test case for controller #2 evaluated the performance robustness of the docking controller against a maximum *negative* starting position uncertainty. So, the chaser was positioned at 42.5 meters behind the docking point at case initiation; this represents a 3-σ (negative) position value from the controller #1 test plan. All other

conditions were held at nominal values throughout the run—including chaser mass, chaser moments of inertia, and initial relative attitude. There was also sensor noise from VisNav and the Kalman filter, which was present in all test cases.

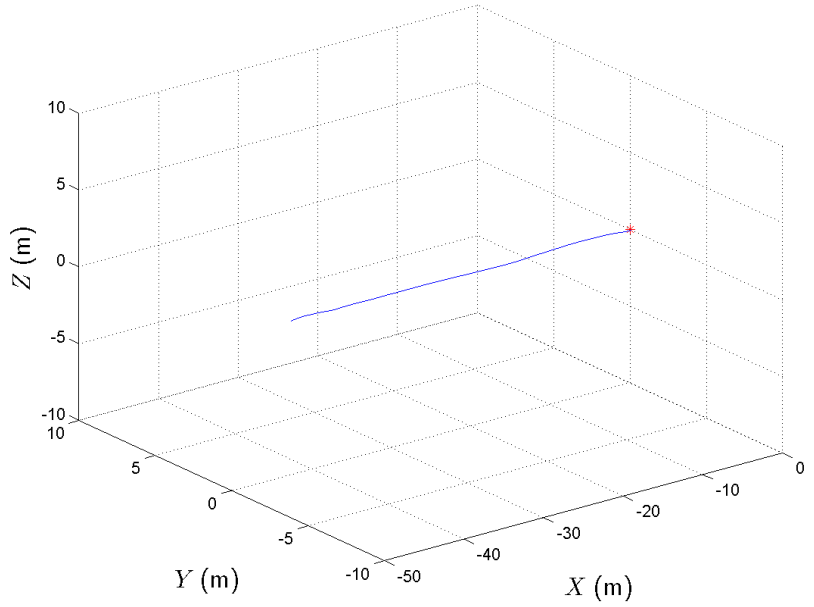

Fig. 62. Chaser Relative Trajectory, Min Position Case

The target frame relative trajectory for the "min position" case is shown in Figure 62. Note that its behavior looks very similar to the nominal just as the "max position" one did; the controller seems to handle the minimum 3-σ position error as well as it did maximum 3-σ position error. This shows an even clearer difference between controller #1 and controller #2, since controller #1 could not have handled this case either.

The Test Case 2 relative state time histories are shown in Figures 63 and 64. Again, these plots and the nominal plots have very similar characteristic behavior, in spite of the difference in initial x position. The final total position error for this case is 7.26 cm, while the final attitude errors are -0.01, -0.17, and 0.19 degrees respectively

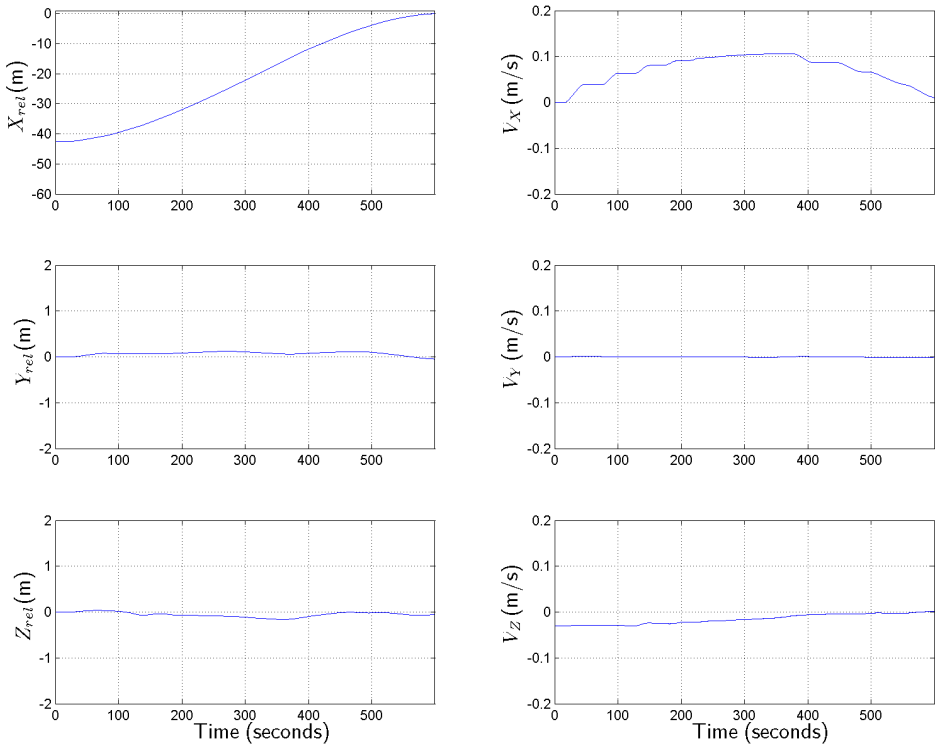

Fig. 63. Chaser Relative Position & Velocity, Min Position Case

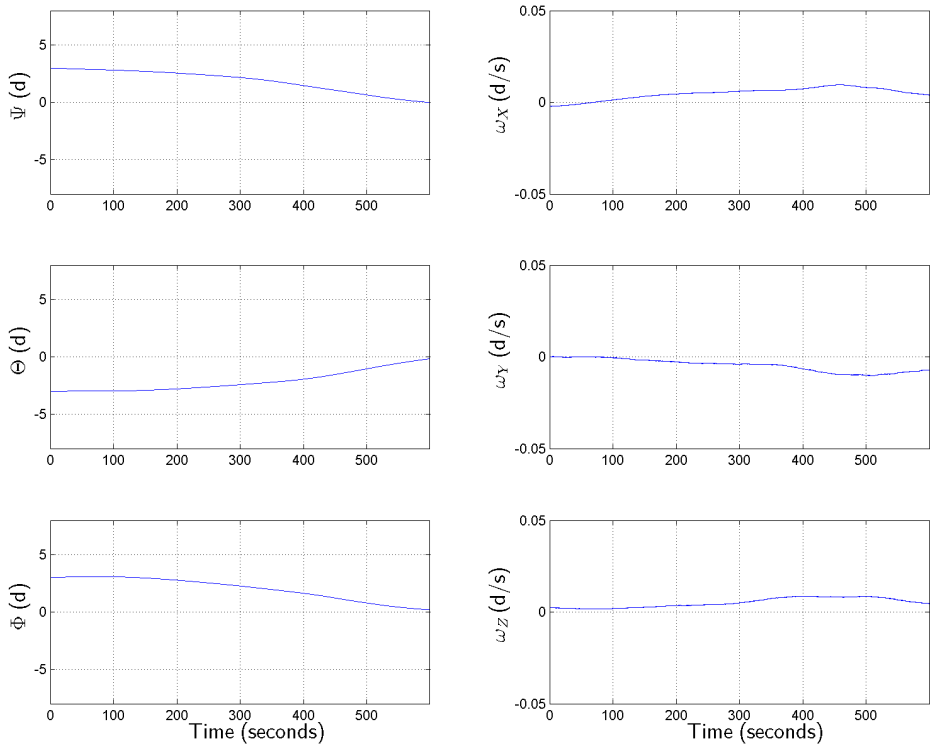

Fig. 64. Chaser Relative Orientation & Attitude Rate, Min Position Case

in yaw, pitch, and roll. These values each clearly eclipsed the required values for a successful dock by a large margin. Finally, the docking was completed at time $t = 599.82$ seconds, which also easily surpassed the time requirement for success.

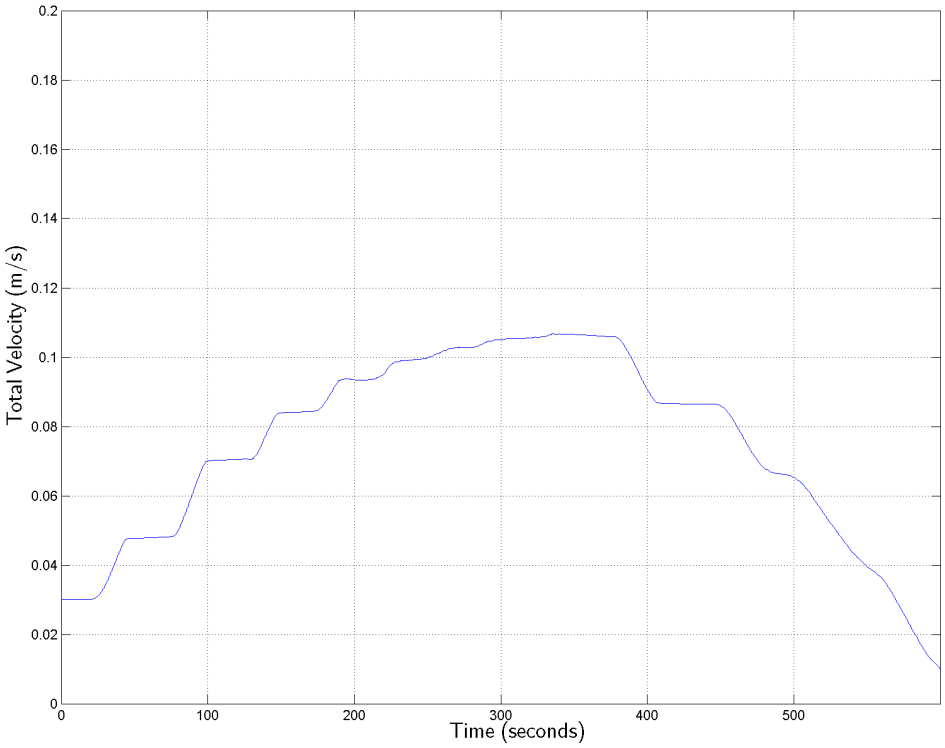

Fig. 65. Relative Velocity Magnitude Profile, Min Position Case

Figure 65 shows a plot of the Test Case 2 total relative velocity magnitude versus time. The graph appears to have slightly less variation in it than Case 1 did, and it of course also shows very desirable general behavior. The velocity magnitude at docking is approximately 1.00 centimeter per second, which is very similar to the nominal case performance. It is also well below both the desired and preferred velocity limits of 10 cm/s and 5 cm/s respectively. These results, combined with the final relative attitude results shown just above, show Test Case 2 to be an "exceptional" run according to

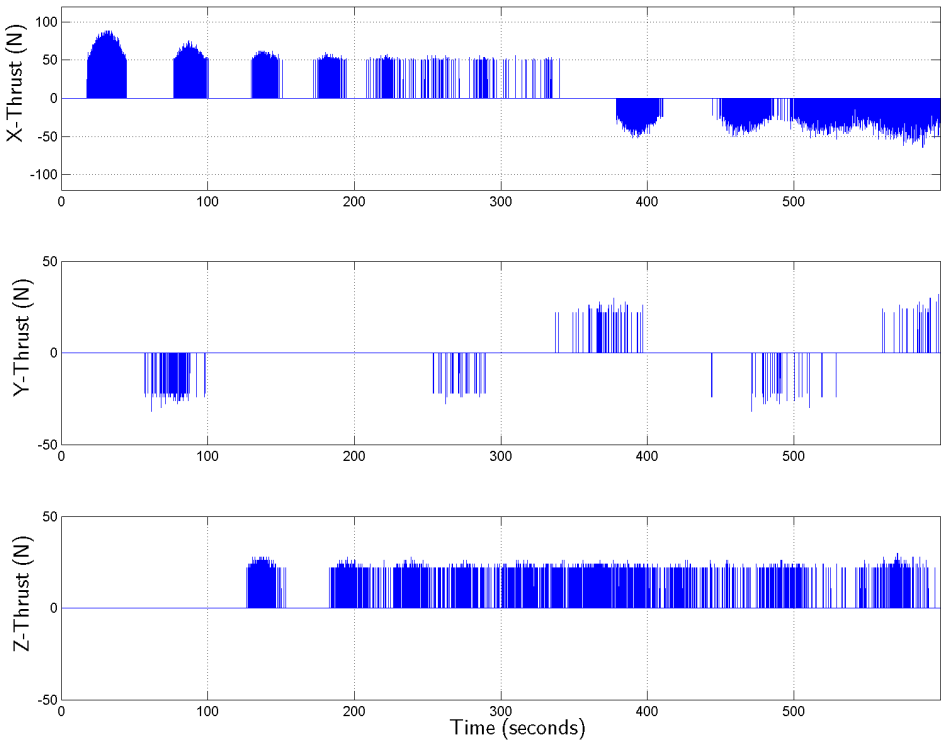

Fig. 66. Chaser Thrust Profile, Min Position Case

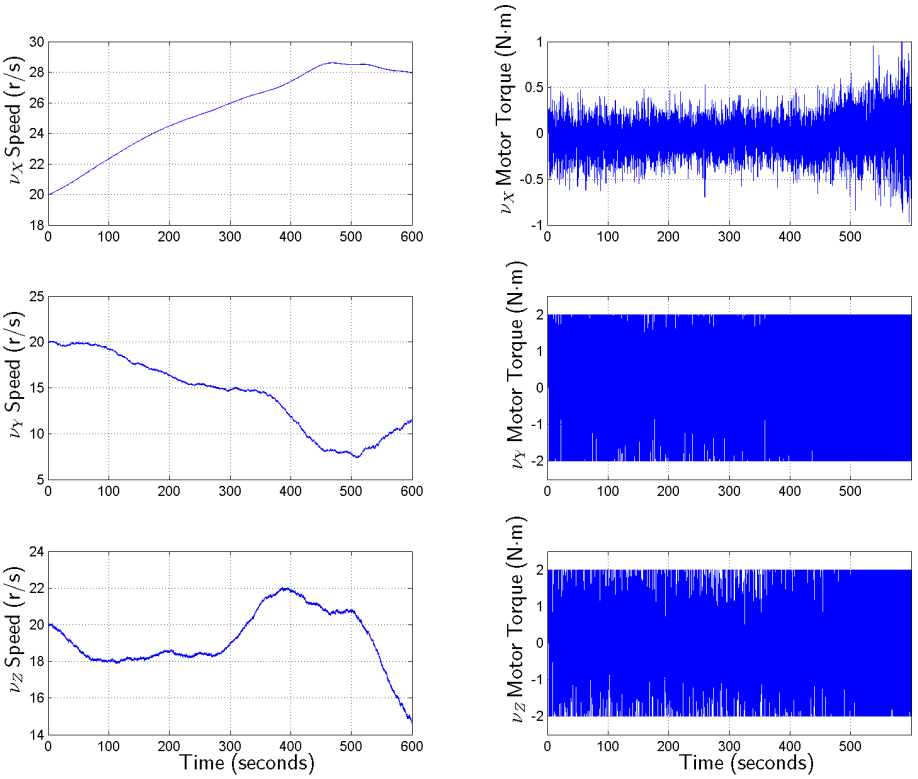

Fig. 67. Wheel Speeds & Motor Torques, Min Position Case

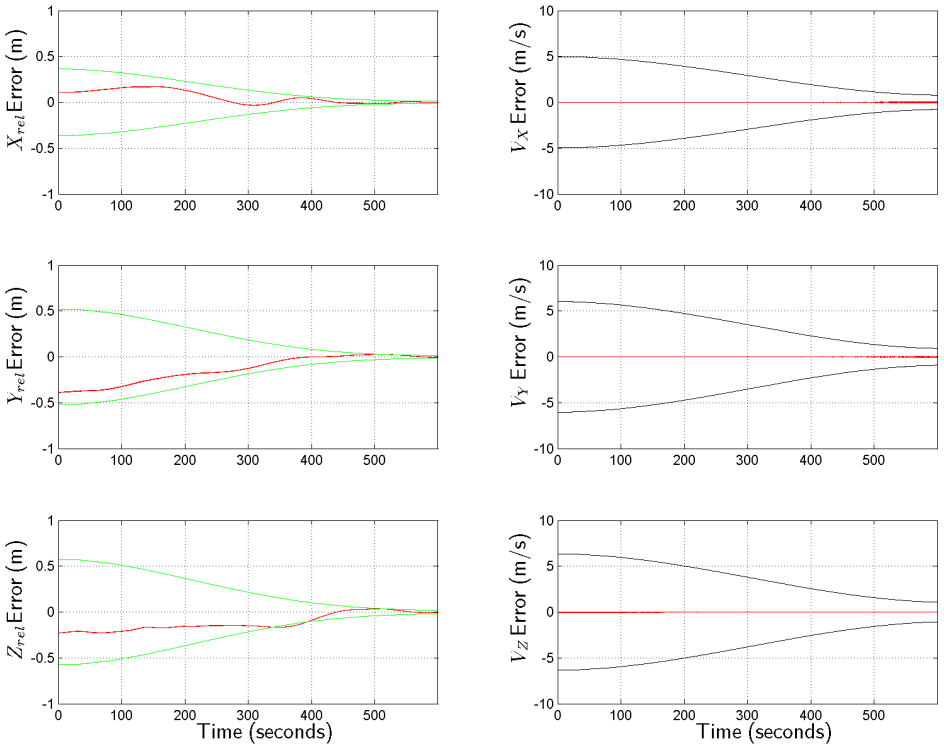

Fig. 68. Position & Velocity Estimate Errors, Min Position Case

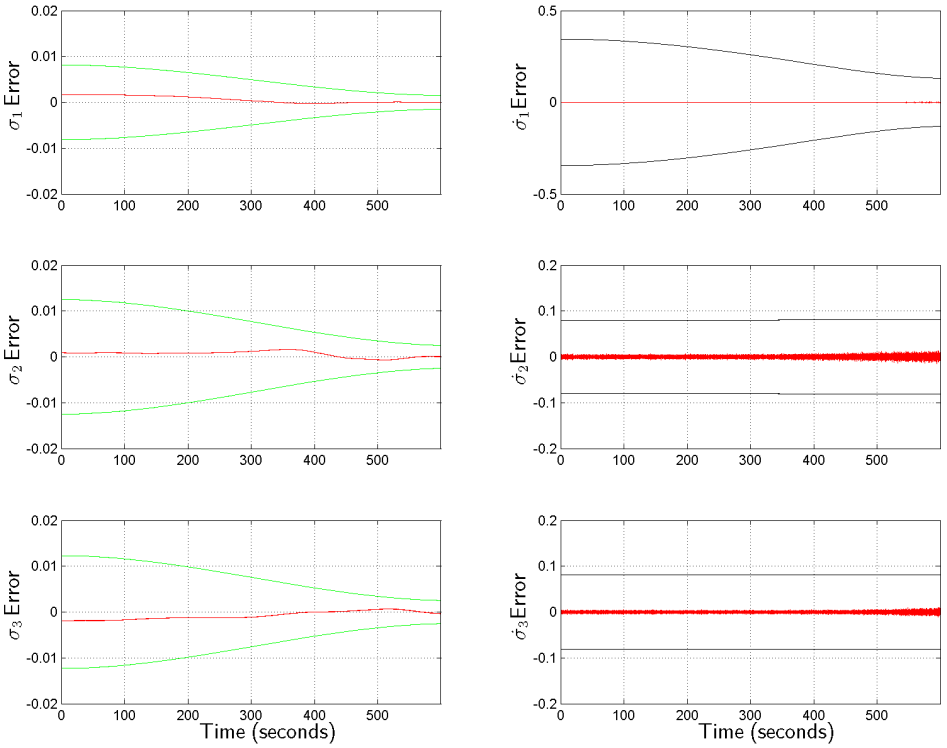

Fig. 69. Orientation & Attitude Rate Estimate Errors, Min Position Case

the criteria for that designation in this experiment.

The rest of the results for the "min position" test case for controller #2 are presented in Figures 66 through 69. Just as for the "max position" case, the graphs are so similar to the nominal ones that they speak for themselves. The controller easily absorbed the negative 3-σ position error initial condition and showed exceptional results in spite of it. Since the VisNav sensor and Kalman filter also performed well, the test case should generally be considered a success on all counts.

These test case results will be further discussed in Sub-section 14, along with the rest of the controller #2 test runs.

4. Test Case 3: Max-Max-Max Inertia Uncertainty

Test Case 3 for controller #2 evaluated the performance robustness of the docking controller against maximum *positive* moment of inertia error in all three axes simultaneously. In the controller #1 test plan the per-axis 3-σ value for inertia variation was 15%; so for consistency, this test case added 15% to the value of each component of the (diagonal) principal inertia matrix. All other conditions were held at nominal values throughout the run—including chaser mass, initial relative attitude, and initial starting position. Sensor noise due to VisNav and the Kalman filter were also present as usual.

The "max-max-max inertia" case target frame relative trajectory is in Figure 70. The shape of the trajectory again strongly resembles the nominal trajectory, as it has for the preceding cases. Controller #2 does not seem to have any difficulty successfully dealing with maximum positive 3-σ inertia errors in all axes simultaneously. This is a positive sign for its performance in general, since this is a worst on worst on worst case that is statistically extremely unlikely and yet it was still able to handle it.

The relative state time histories for this test case are shown in Figures 71 and 72.

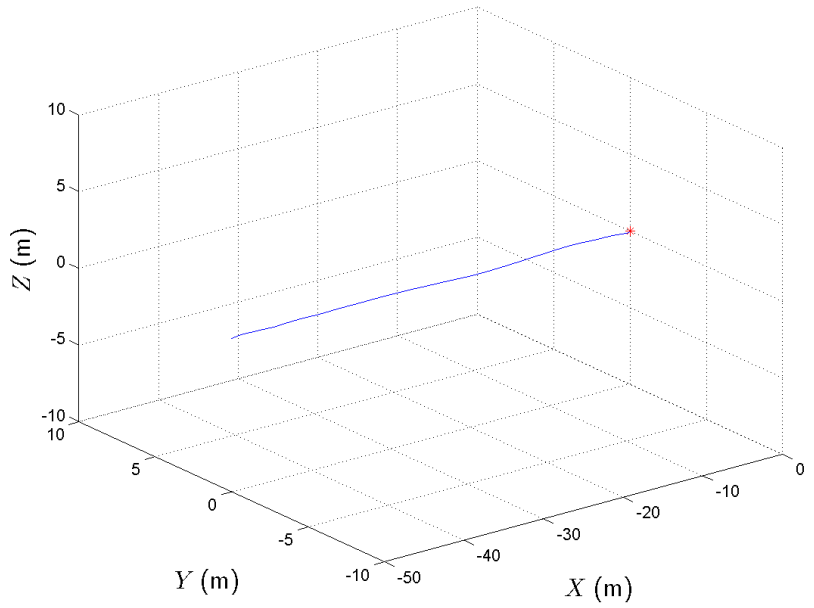

Fig. 70. Chaser Relative Trajectory, Max-Max-Max Inertia Case

The graphs are consistent with the trajectory plot in showing very similar performance to that of the nominal controller #2 case. The total position error at docking for this case is 5.61 cm, and the final errors in yaw, pitch, and roll are -0.05, -0.31, and 0.18 degrees respectively. Docking occurred at 601.10 seconds in this run. Note that all of these values easily surpassed their respective requirements for the run to be considered a successful docking.

The total relative velocity magnitude versus time for Test Case 3 is plotted in Figure 73. The graph shows very desirable general behavior, which is also similar to the nominal case. The docking velocity magnitude is also very similar to the nominal value at about 1.05 centimeter per second; this easily clears both the desired and preferred velocity limits (10 cm/s and 5 cm/s respectively). In short, this test case is considered an "exceptional" one based on its excellent final velocity and final attitude error results, which cleared all of the preferred criteria with ease.

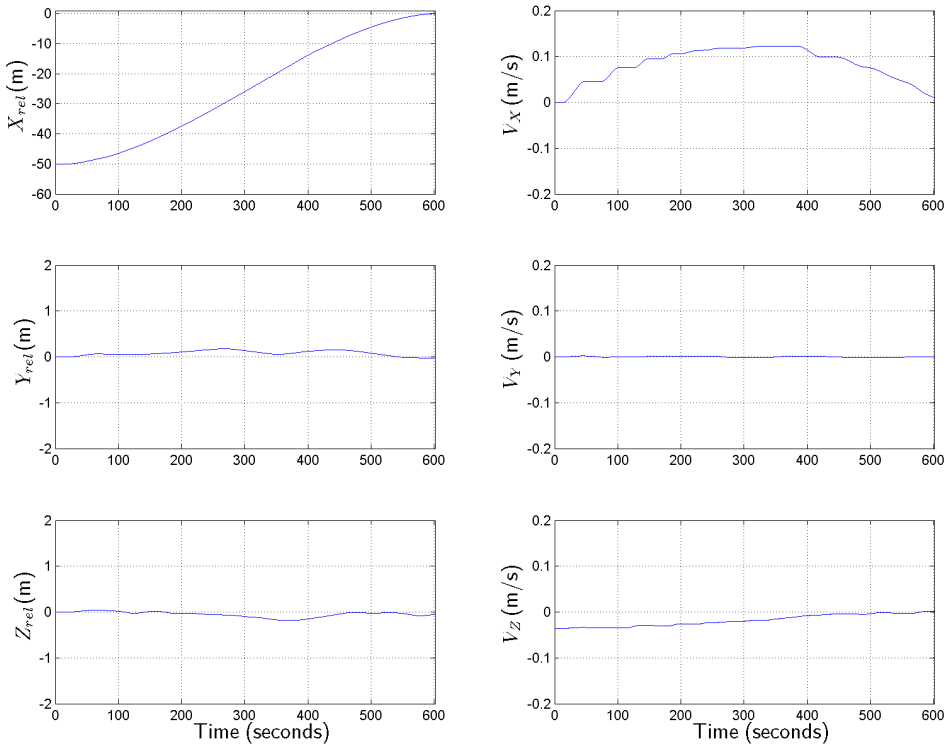

Fig. 71. Chaser Relative Position & Velocity, Max-Max-Max Inertia Case

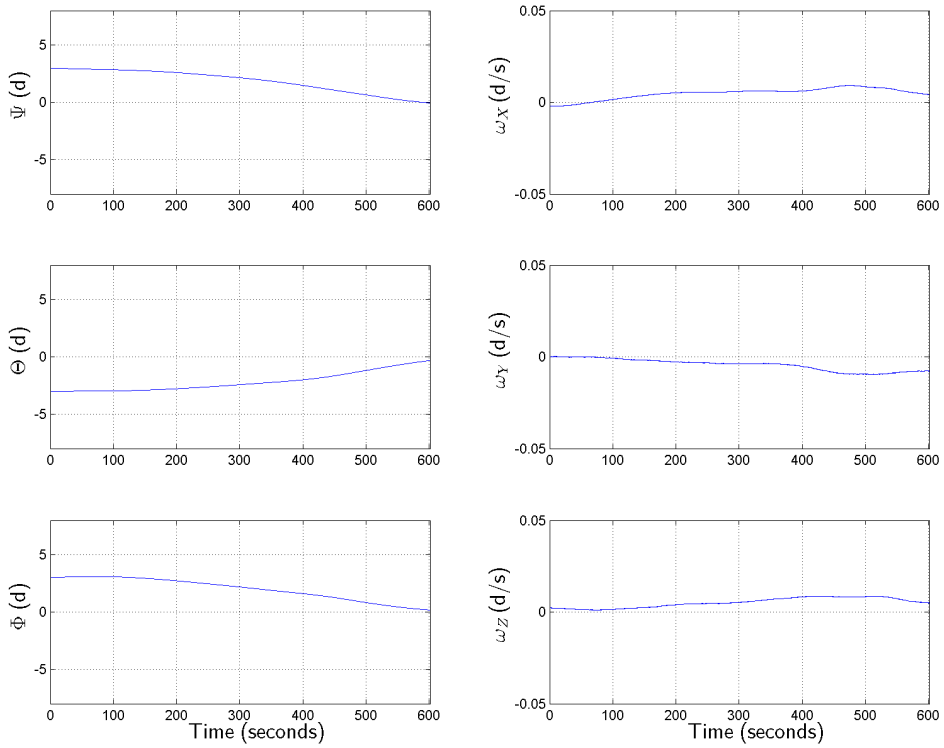

Fig. 72. Chaser Relative Orientation & Attitude Rate, Max-Max-Max Inertia Case

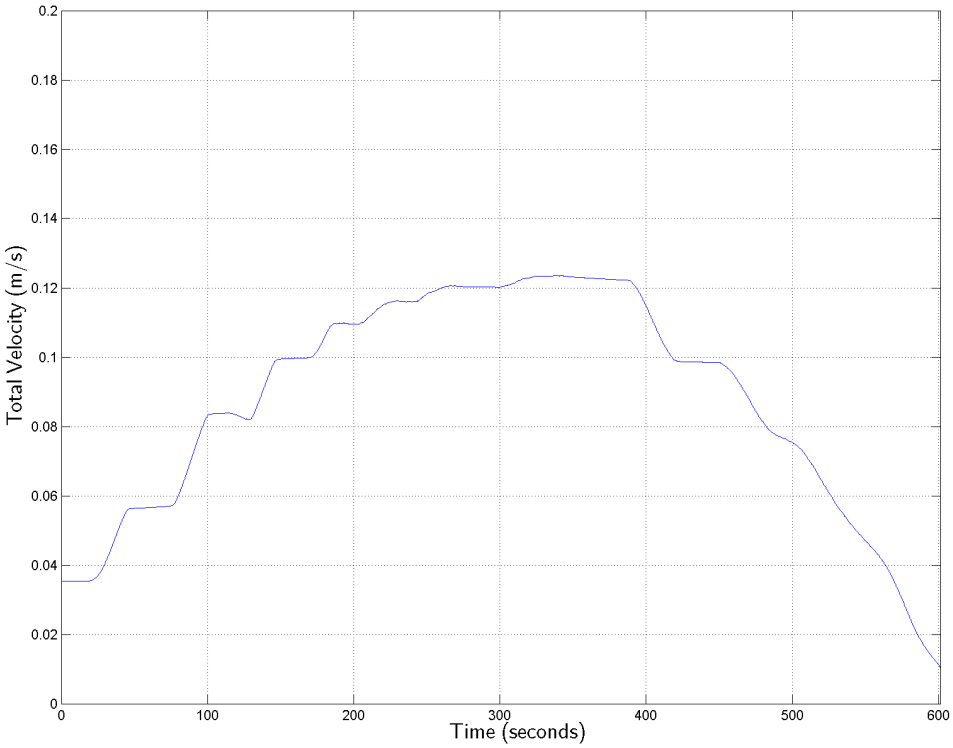

Fig. 73. Relative Velocity Magnitude Profile, Max-Max-Max Inertia Case

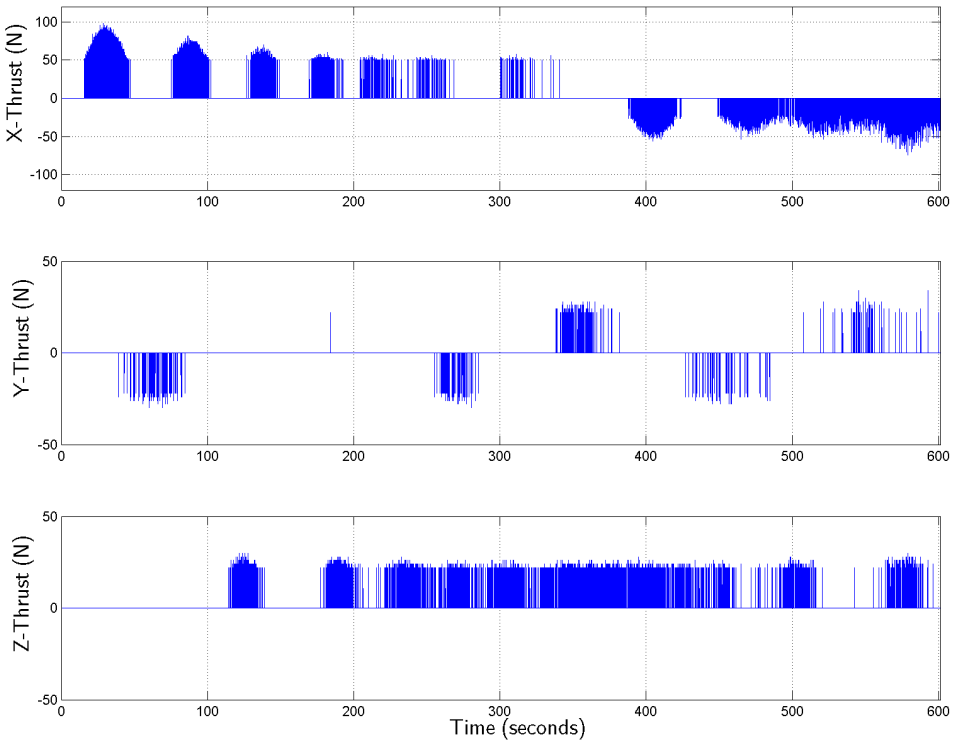

Fig. 74. Chaser Thrust Profile, Max-Max-Max Inertia Case

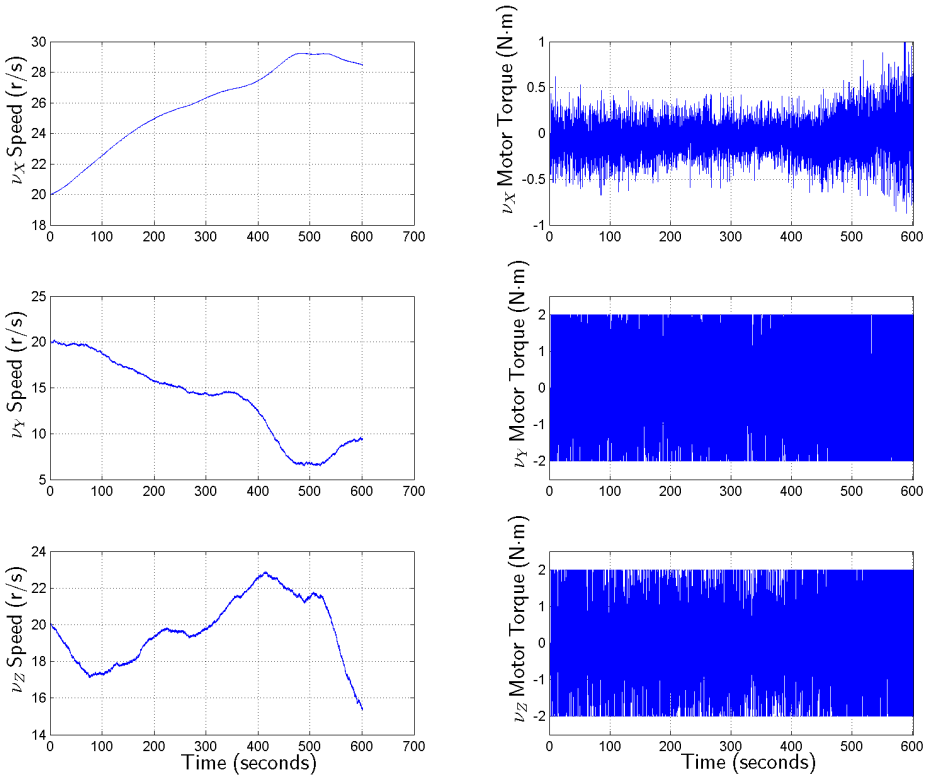

Fig. 75. Wheel Speeds & Motor Torques, Max-Max-Max Inertia Case

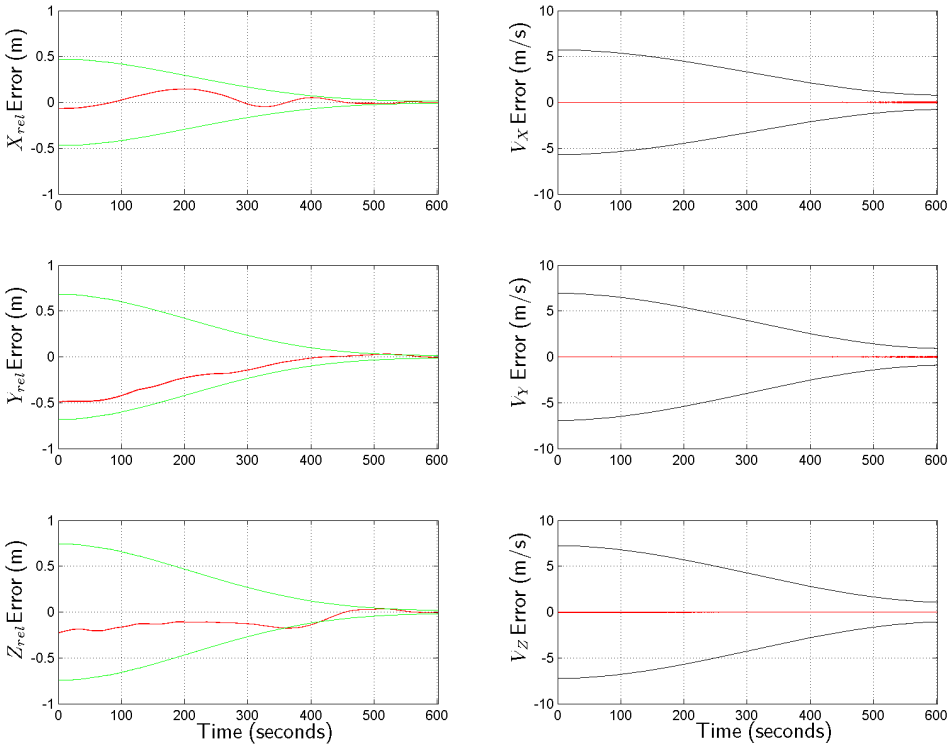

Fig. 76. Position & Velocity Estimate Errors, Max-Max-Max Inertia Case

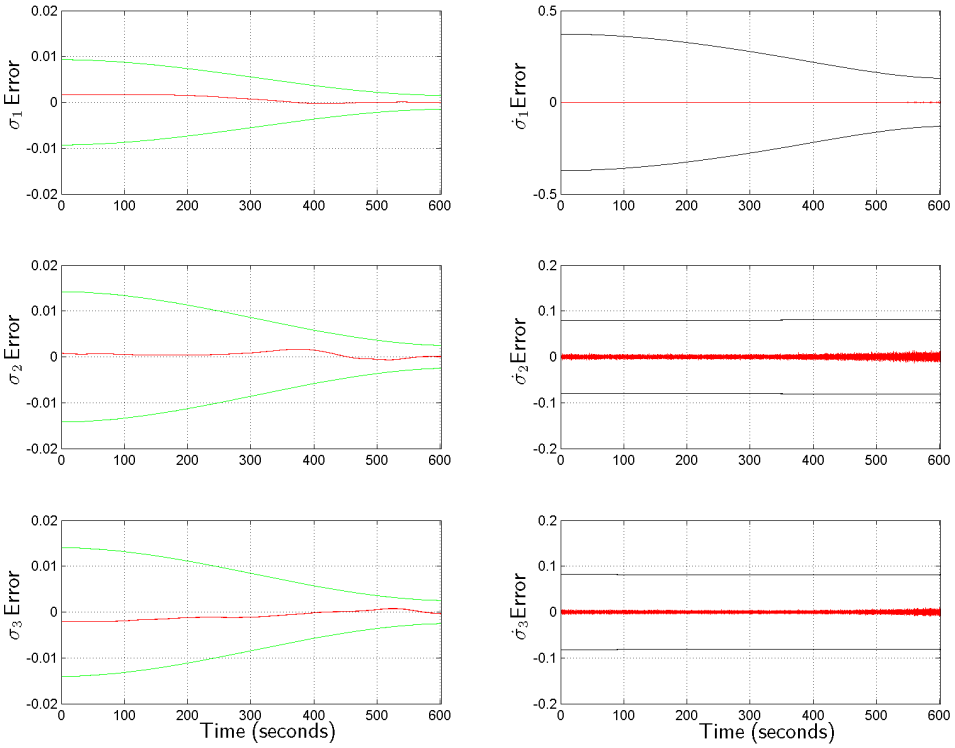

Fig. 77. Orientation & Attitude Rate Estimate Errors, Max-Max-Max Inertia Case

Figures 74 through 77 detail the rest of the results for the "max-max-max inertia" controller #2 test case. The plots are simply too similar to what has already been seen in the previous cases, including the nominal, to warrant any dedicated discussion. The controller, as well as VisNav and the Kalman filter, performed extremely well in this test case, acommodating the statistically unlikely worst on worst on worst maximum positive inertia errors without any problem; this fact is borne out in the graphs.

The results for this test case will be evaluated in light of the rest of the controller #2 test runs in Sub-section 14.

5. Test Case 4: Min-Min-Min Inertia Uncertainty

The fourth controller #2 test case examined its performance robustness against maximum *negative* moment of inertia error in all three axes simultaneously. Similar to the third test case, the per-axis 3-σ value for inertia variation from the controller #1 test plan was used, except that the principal moment of inertia value in each axis was *reduced* by 15% for this run. All other conditions were given nominal values for the run—including chaser mass, initial relative attitude, and initial starting position. The usual sensor noise due to VisNav and the Kalman filter was also present.

Figure 78 shows the chaser vehicle's relative trajectory in target frame coordinates. Based on the shape of the path, it appears that the controller's performance in this case was again similar to its nominal case behavior. It seems to have easily dealt with the maximum negative 3-σ inertia errors in all axes simultaneously, in spite of that being a difficult and statistically unlikely scenario.

The time histories of all of the relative states for this "min-min-min inertia" test case are presented in Figures 79 and 80. These plots too indicate that the controller's performance was very similar to that of the nominal case. The final total position error

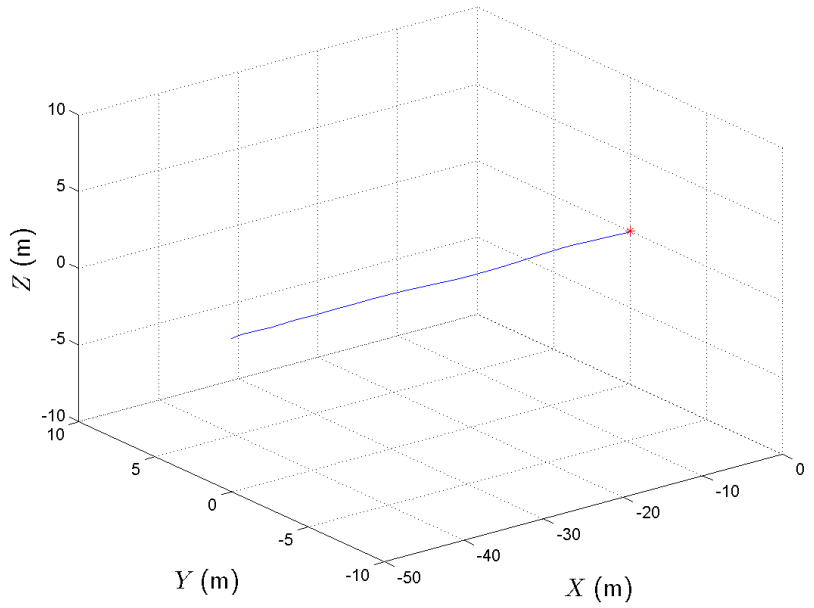

Fig. 78. Chaser Relative Trajectory, Min-Min-Min Inertia Case

was 6.15 cm, and the errors in yaw, pitch, and roll at docking were 0.02, -0.19, and 0.19 degrees respectively. The manuever completed in 601.46 seconds for this case. Each of these values easily eclipsed the requirements for the run to be considered a successful docking, while the final attitude errors amply cleared the "preferred" criteria as well.

The Test Case 4 total relative velocity magnitude time history is displayed in Figure 81. Similar to the nominal case, the total velocity exhibits very desirable general behavior—especially at docking. In fact, the velocity magnitude at docking was actually better than the nominal at about 0.89 centimeter per second. This value surpasses both the desired and preferred limits of 10 cm/s and 5 cm/s for final velocity. Thus, this test case is "exceptional", since the final attitude errors also exceeded their respective "preferred" criteria.

Figures 82 through 85 show the remaining results plots for this test case. The

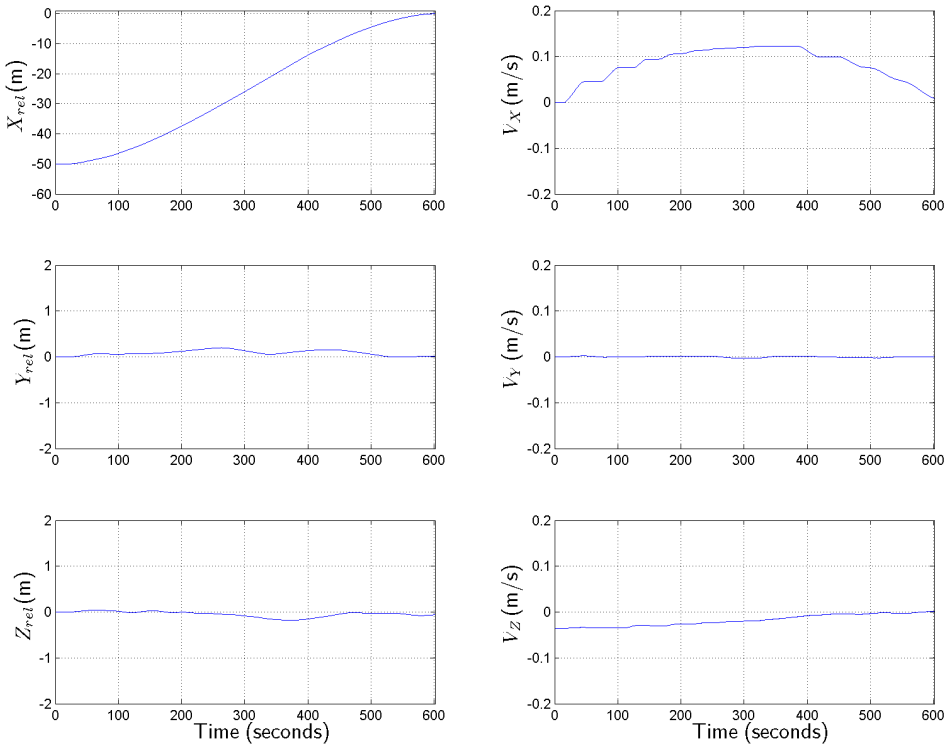

Fig. 79. Chaser Relative Position & Velocity, Min-Min-Min Inertia Case

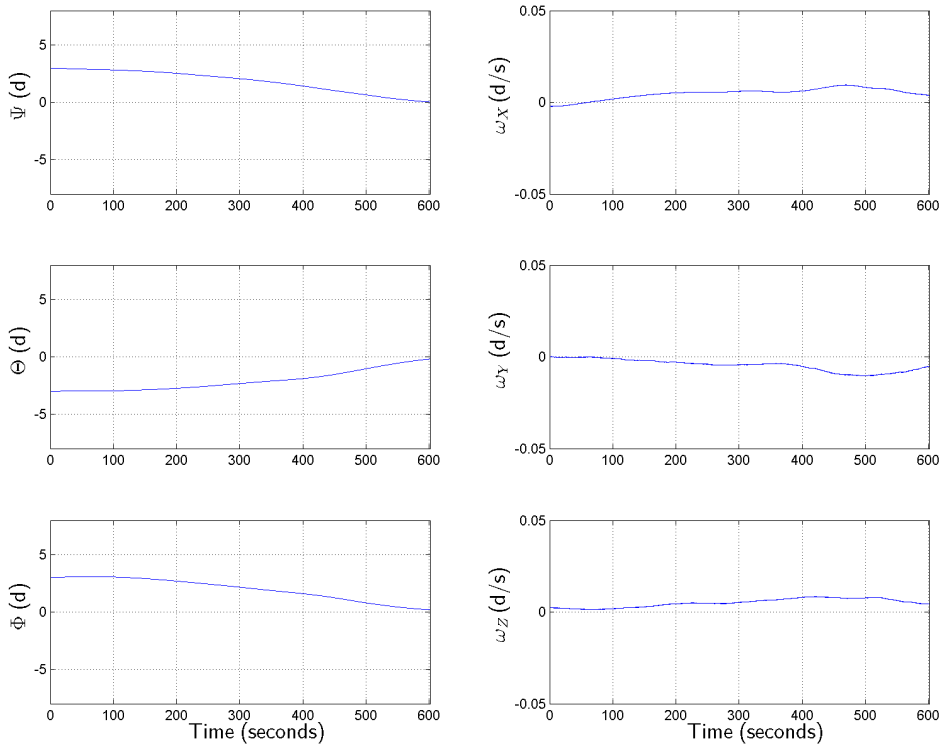

Fig. 80. Chaser Relative Orientation & Attitude Rate, Min-Min-Min Inertia Case

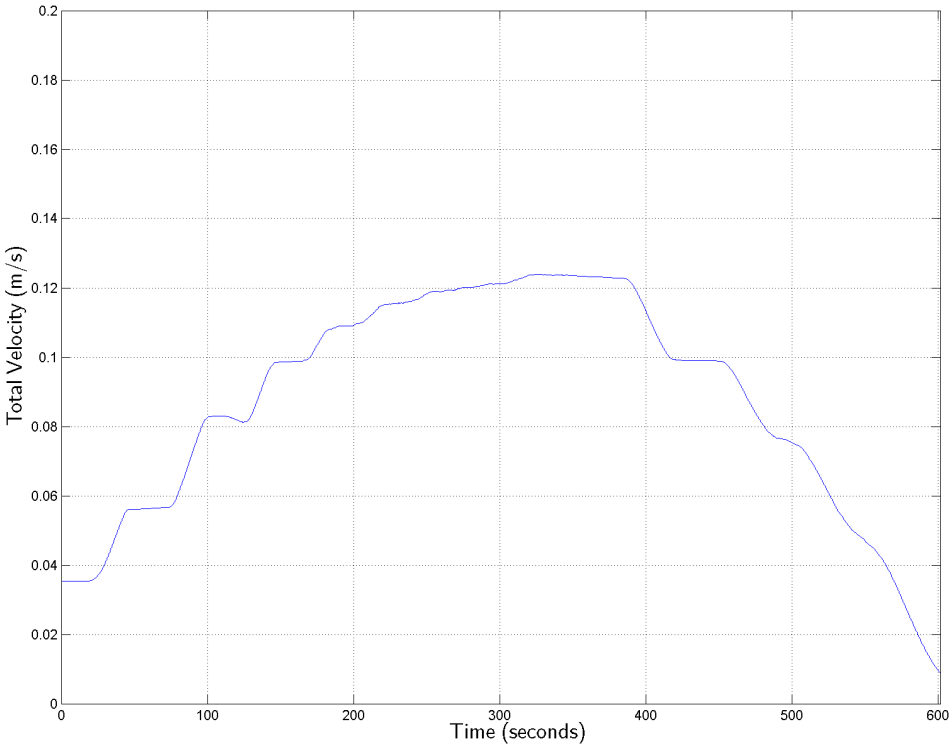

Fig. 81. Relative Velocity Magnitude Profile, Min-Min-Min Inertia Case

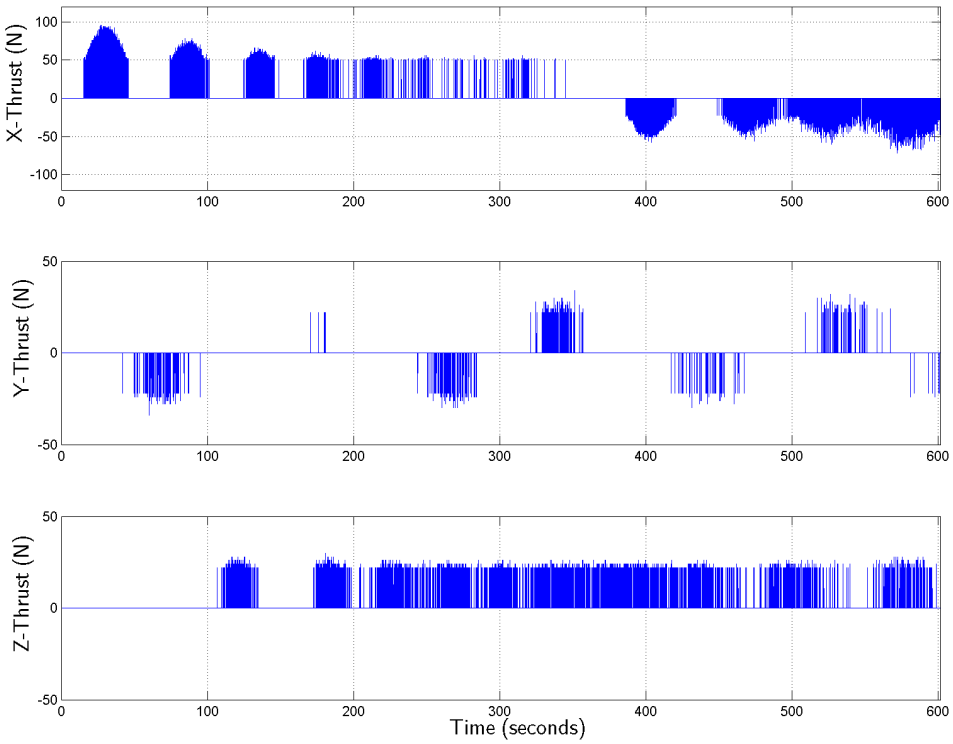

Fig. 82. Chaser Thrust Profile, Min-Min-Min Inertia Case

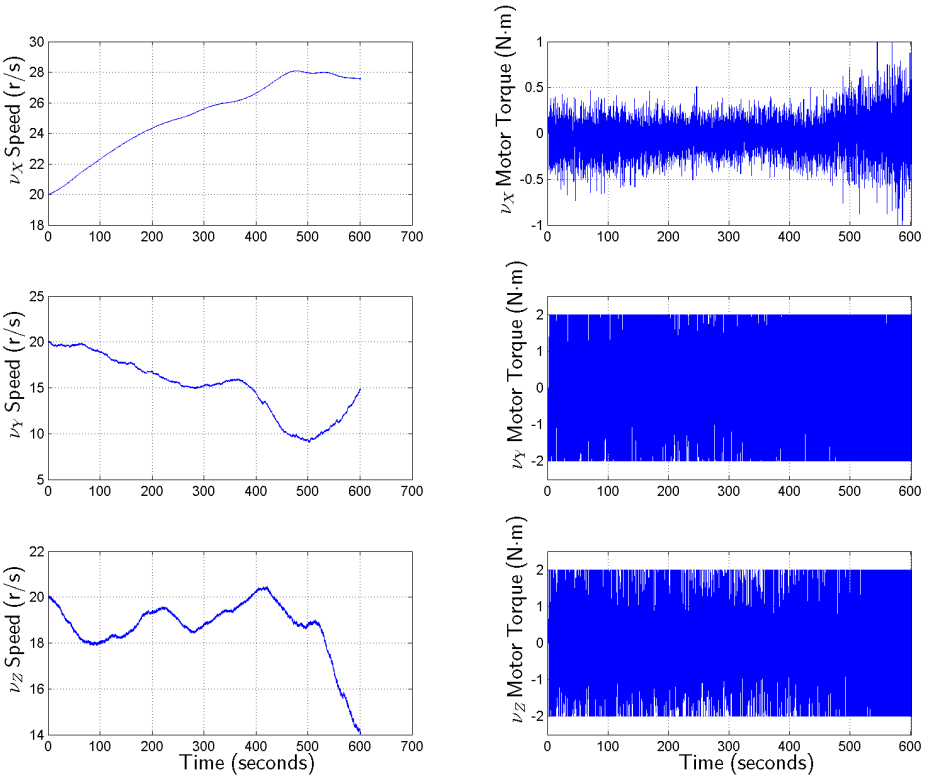

Fig. 83. Wheel Speeds & Motor Torques, Min-Min-Min Inertia Case

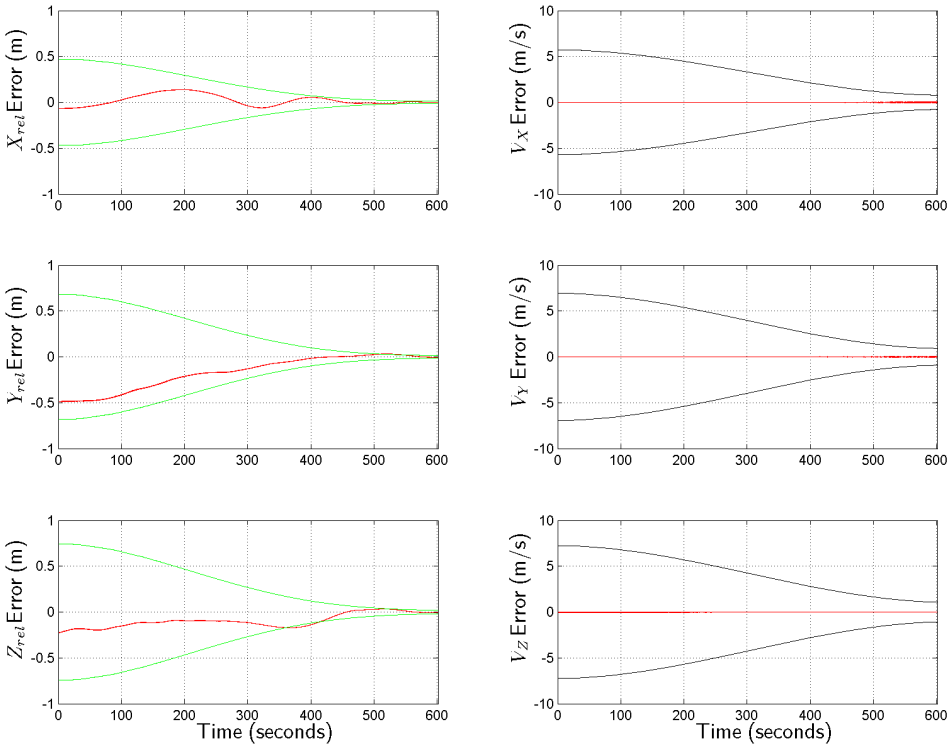

Fig. 84. Position & Velocity Estimate Errors, Min-Min-Min Inertia Case

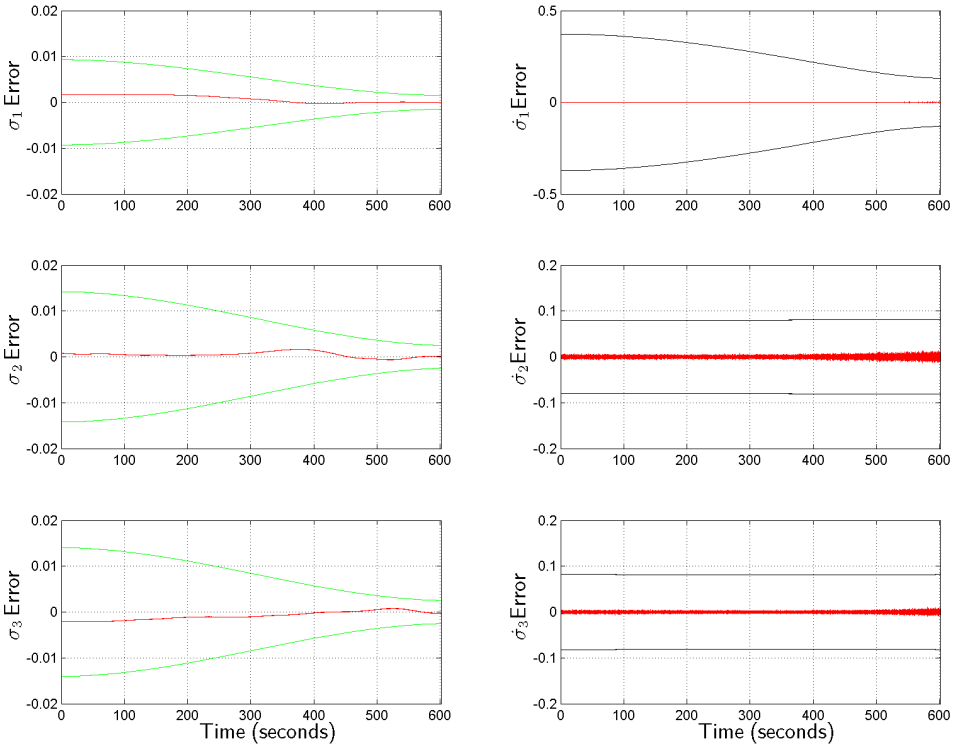

Fig. 85. Orientation & Attitude Rate Estimate Errors, Min-Min-Min Inertia Case

plots are similar enough in behavior to the previous cases, including nominal, that they warrant no particular discussion. Basically, controller #2, VisNav, and the Kalman filter all performed quite well throughout this test case, handling the statistically unlikely worst on worst on worst maximum negative inertia errors without issue; the remaining plots serve as sufficient evidence to this fact.

Test case 4 results will be analyzed in the context of the entire set of controller #2 test cases in Sub-section 14.

6. Test Case 5: Max-Min-Min Inertia Uncertainty

Controller #2's Test Case 5 examined its performance robustness against maximum positive moment of inertia error about the x-axis, simultaneous with maximum negative moment of inertia errors about the y- and z-axes. In all axes, this resulted in a 15% change from the nominal principal inertia value; this is because the per-axis 3-σ values for inertia variation in the controller #1 test plan were $\pm$15%. Everything else in the simulation was held at nominal for the run—including chaser mass, initial relative attitude, and initial starting position. VisNav and Kalman filter sensor noise were also present as for every other case.

Figure 86 shows the relative trajectory of the chaser vehicle for this test case, expressed in target frame coordinates. The general shape of the trajectory appears to indicate that the controller's performance in this case also was similar to its nominal case behavior. This is a very favorable result for the controller, considering the current case is a statistically unlikely worst on worst on worst scenario.

Figures 87 and 88 show the relative state time histories for the current "max-min-min inertia" test case. In these plots it is also readily apparent that the controller's performance was very similar to its nominal case performance. The final total position error for this case was 7.67 cm, with the final attitude errors at values of 0.08, -0.20,

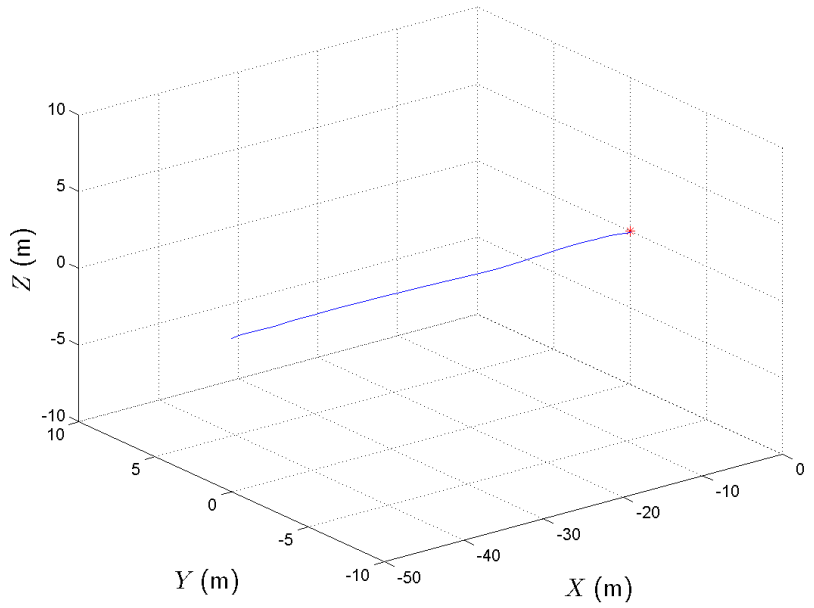

Fig. 86. Chaser Relative Trajectory, Max-Min-Min Inertia Case

and 0.17 degrees respectively for yaw, pitch, and roll. For this case, the entire docking manuever completed in 600.51 seconds. The final values for position, attitude, and time all easily cleared the requirements for a successful docking run, and the final attitude errors also eclipsed the "preferred" criteria by a large amount.

Figure 89 shows a plot of the total relative velocity magnitude time history for Test Case 5. As was seen in the nominal case, this quantity exhibits desirable general behavior at all times, but especially at and near docking. The value of total velocity at the final time was 0.99 centimeters per second, which bettered both the required and preferred criteria (10 cm/s and 5 cm/s respectively) for final velocity. So, this test case registers as "exceptional", since the final attitude was also better than its "preferred" criteria in each axis.

The remaining results plots for this test case are displayed in Figures 90, 91, 92, and 93. Since the plots are so similar to the nominal case, no specific discussion of

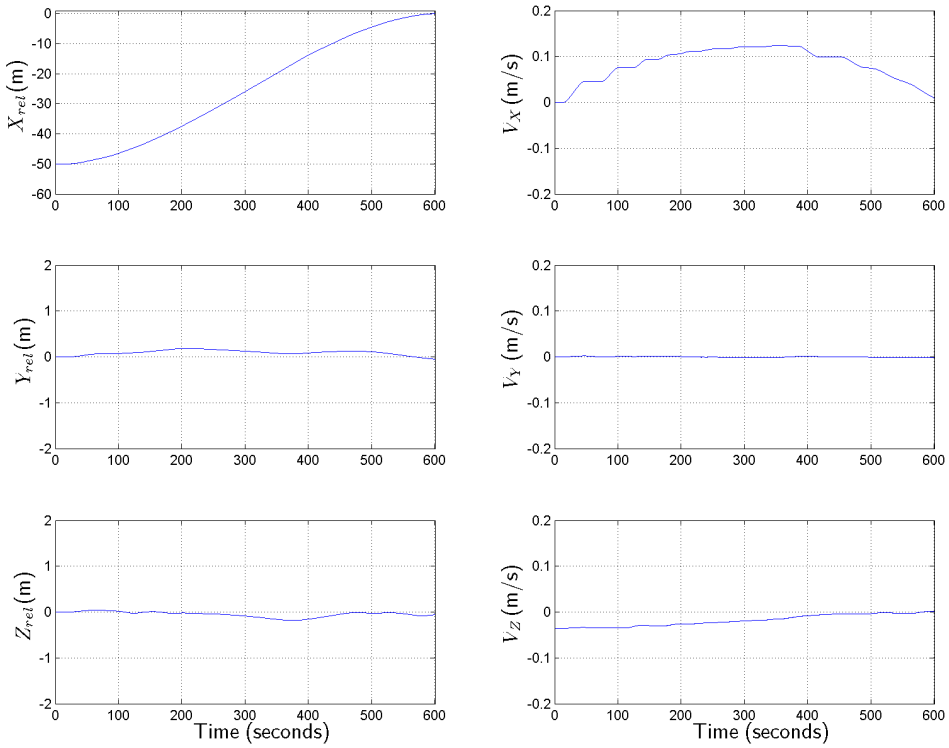

Fig. 87. Chaser Relative Position & Velocity, Max-Min-Min Inertia Case

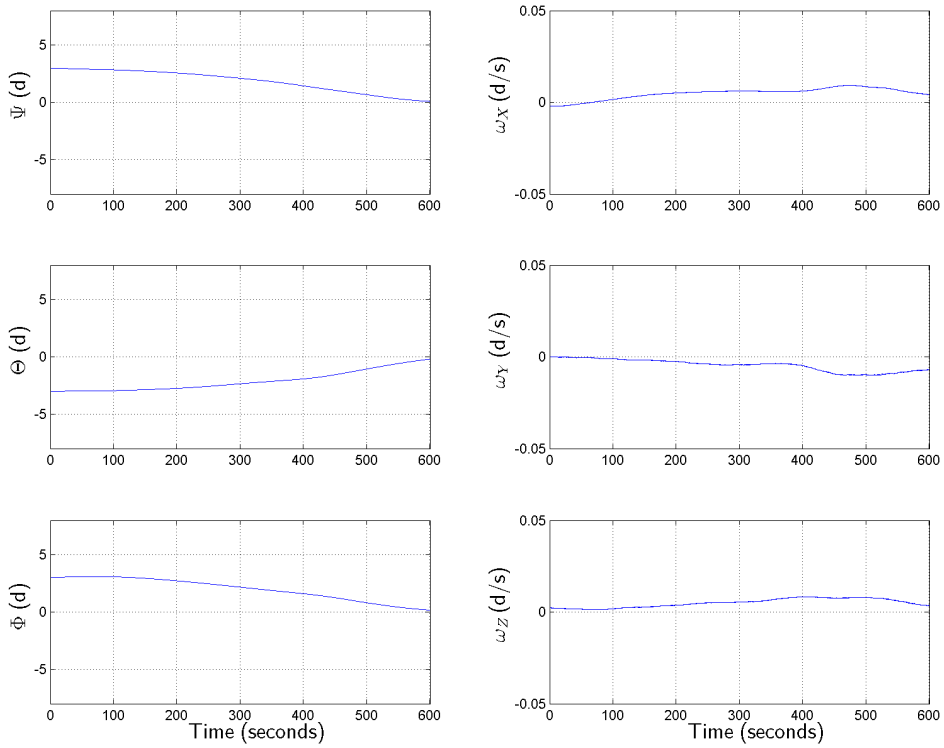

Fig. 88. Chaser Relative Orientation & Attitude Rate, Max-Min-Min Inertia Case

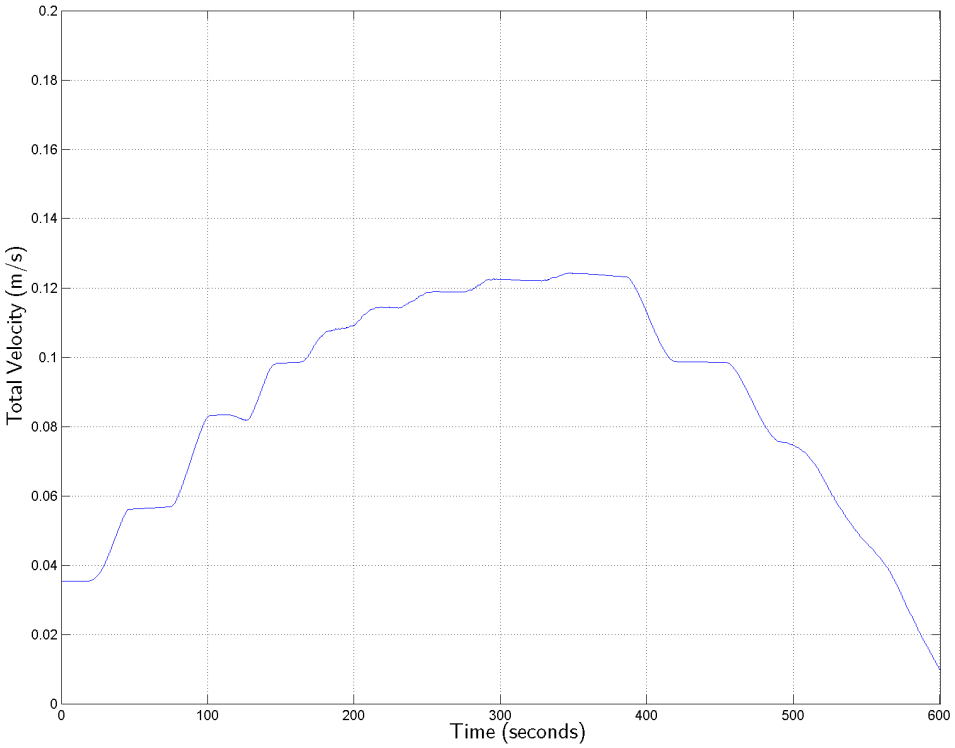

Fig. 89. Relative Velocity Magnitude Profile, Max-Min-Min Inertia Case

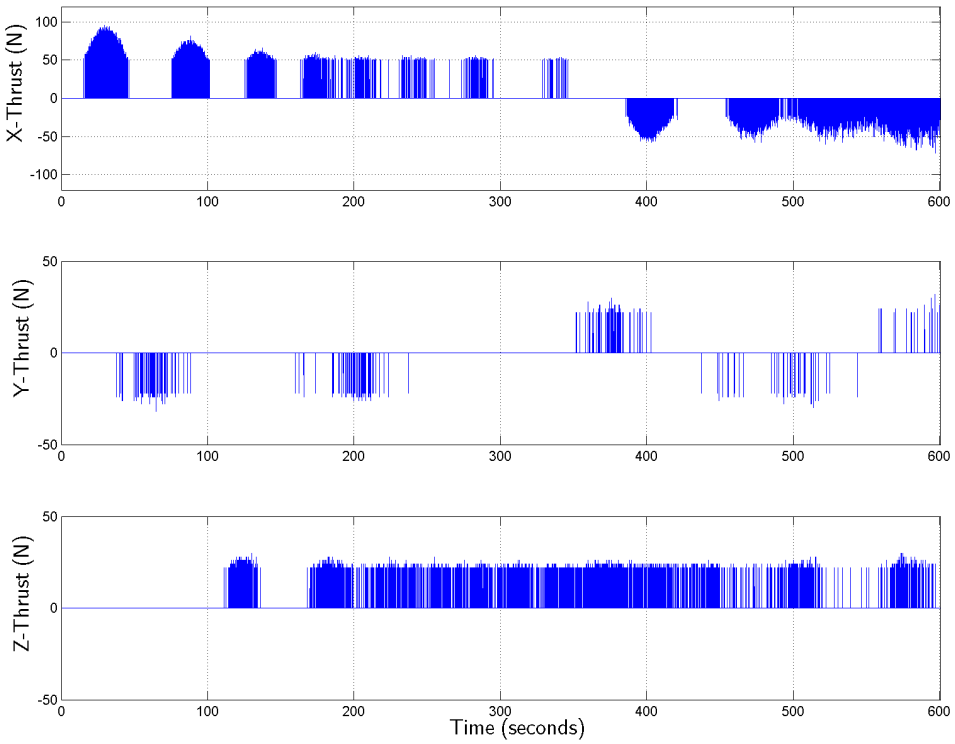

Fig. 90. Chaser Thrust Profile, Max-Min-Min Inertia Case

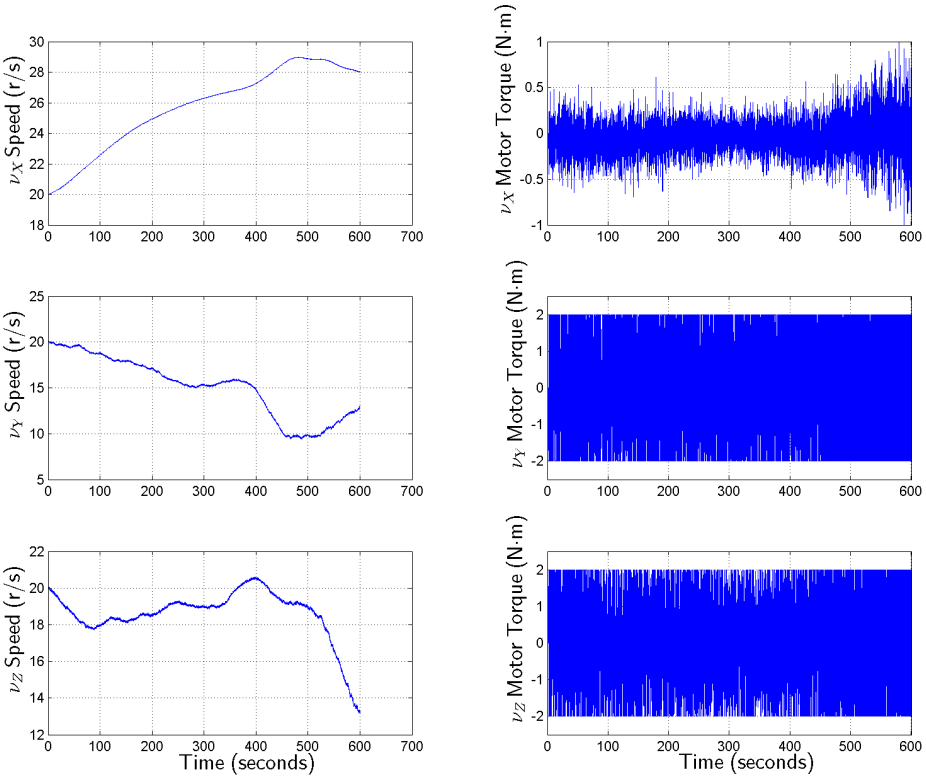

Fig. 91. Wheel Speeds & Motor Torques, Max-Min-Min Inertia Case

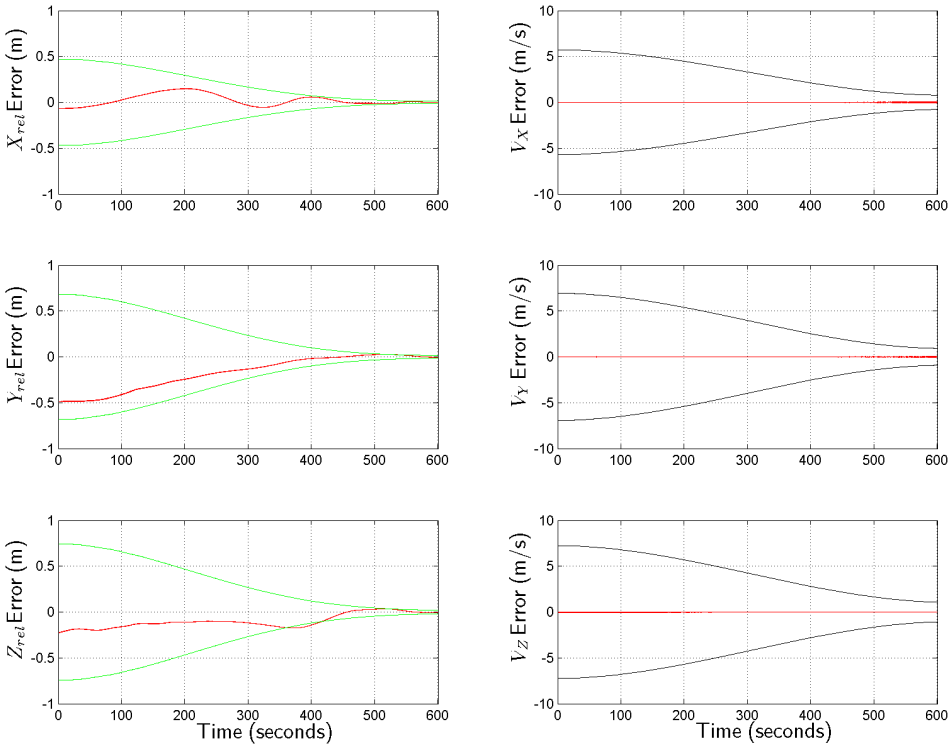

Fig. 92. Position & Velocity Estimate Errors, Max-Min-Min Inertia Case

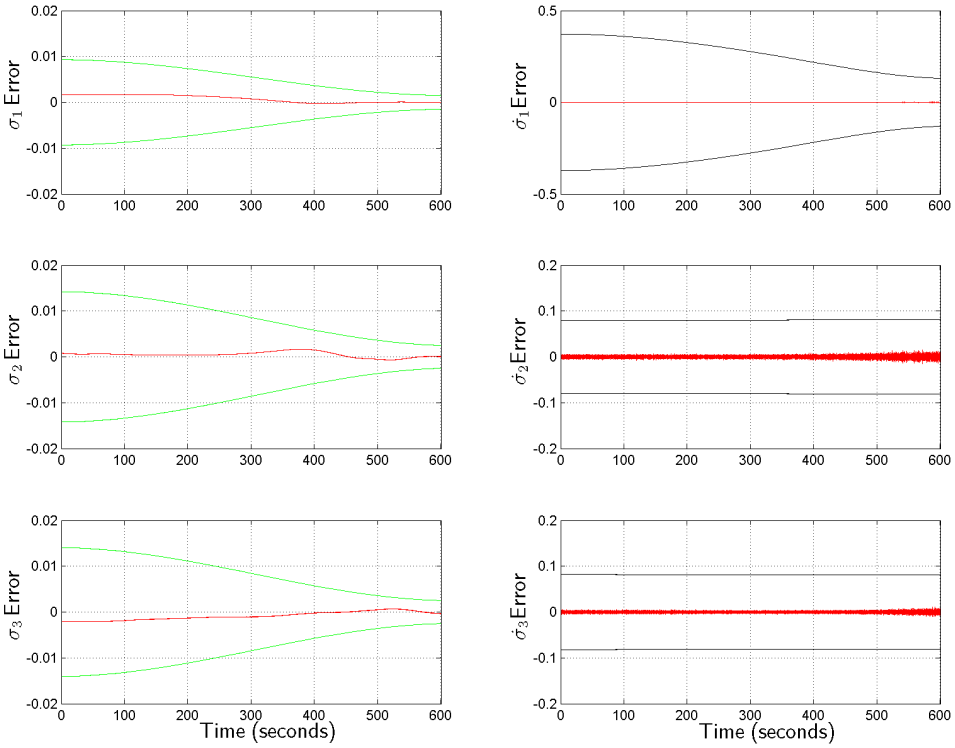

Fig. 93. Orientation & Attitude Rate Estimate Errors, Max-Min-Min Inertia Case

them is deemed necessary here. To summarize them, the performance of both the controller and the VisNav-Kalman filter combination was excellent for the duration of this run, in spite of the fact that the scenario represented a statistically highly unlikely worst on worst on worst maximum inertia error case.

The results for this test case will be further discussed, along with those of the other controller #2 test cases, in Sub-section 14.

7. Test Case 6: Min-Max-Max Inertia Uncertainty

The performance of controller #2 was evaluated against maximum negative moment of inertia error about the x-axis, simultaneous with maximum positive moment of inertia errors about the y- and z-axes, in Test Case 6. These values were a 15% change from the nominal principal inertias in each axis; 15% was used for this test because it was the chosen per-axis 3-σ values for inertia variation in the controller #1 test plan. All other simulation parameters were held at nominal values for the duration of the run—including chaser mass, initial relative attitude, and initial starting position. Sensor noise due to VisNav and the Kalman filter were also part of the scenario as for every case.

Figure 94 shows the chaser vehicle relative trajectory for the current case as expressed in target frame coordinates. The plot indicates that the controller performed very well in this case, since the trajectory appears to look generally very like the nominal case trajectory. Since the current test case is a statistically unlikely worst on worst on worst condition, this performance reflects well on the controller's robustness to moment of inertia errors—particularly in light of similarly positive results in the three preceding inertia extremal cases.

The relative state plots with respect to time for this case are shown in Figures 95 and 96. These plots also show that the controller performed very similarly to the

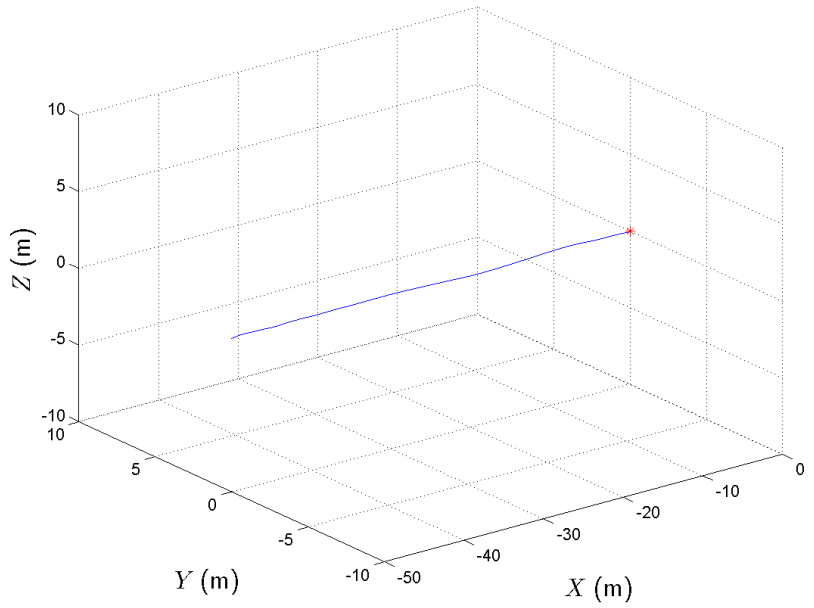

Fig. 94. Chaser Relative Trajectory, Min-Max-Max Inertia Case

nominal in the current "min-max-max inertia" case. The position error at docking for Test Case 6 was 7.51 cm, while the final attitude errors were -0.01, -0.19, and 0.21 degrees respectively (yaw, pitch, roll order). Final docking time for this case was 600.62 seconds. According to the proscribed success criteria, the values for position, attitude, and time at docking were all acceptable, with the final attitude errors even meeting the "preferred" criteria very easily.

The Test Case 6 total relative velocity magnitude versus time is plotted in Figure 97. It is apparent from the figure that the total velocity behaves desirably for the duration of the test run, especially during the endgame when it is most important. The final total velocity was 0.90 centimeters per second for the current test case; this value exceeded even the preferred docking velocity criteria of 5 cm/s quite easily. That, plus the excellent behavior in attitude that also exceeded the preferred criteria, causes this test case to be considered an "exceptional" one.

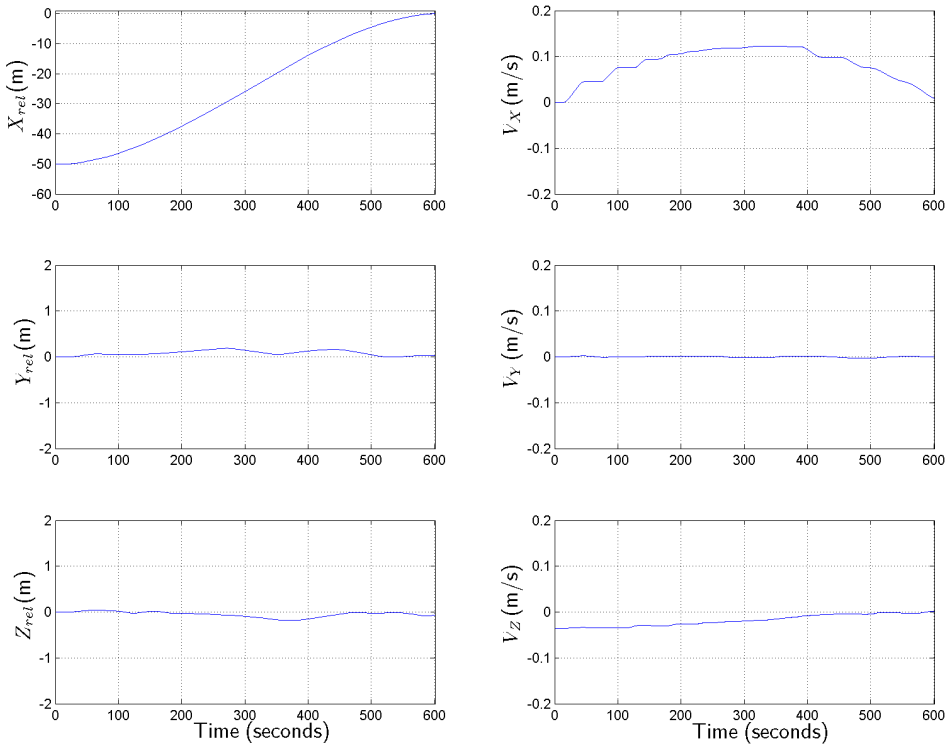

Fig. 95. Chaser Relative Position & Velocity, Min-Max-Max Inertia Case

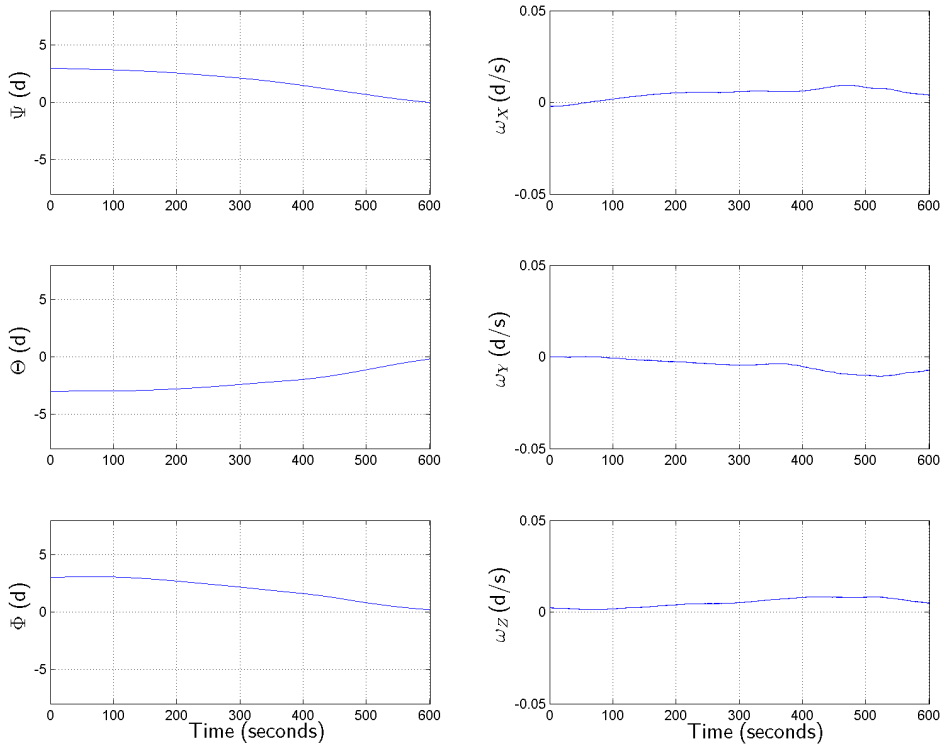

Fig. 96. Chaser Relative Orientation & Attitude Rate, Min-Max-Max Inertia Case

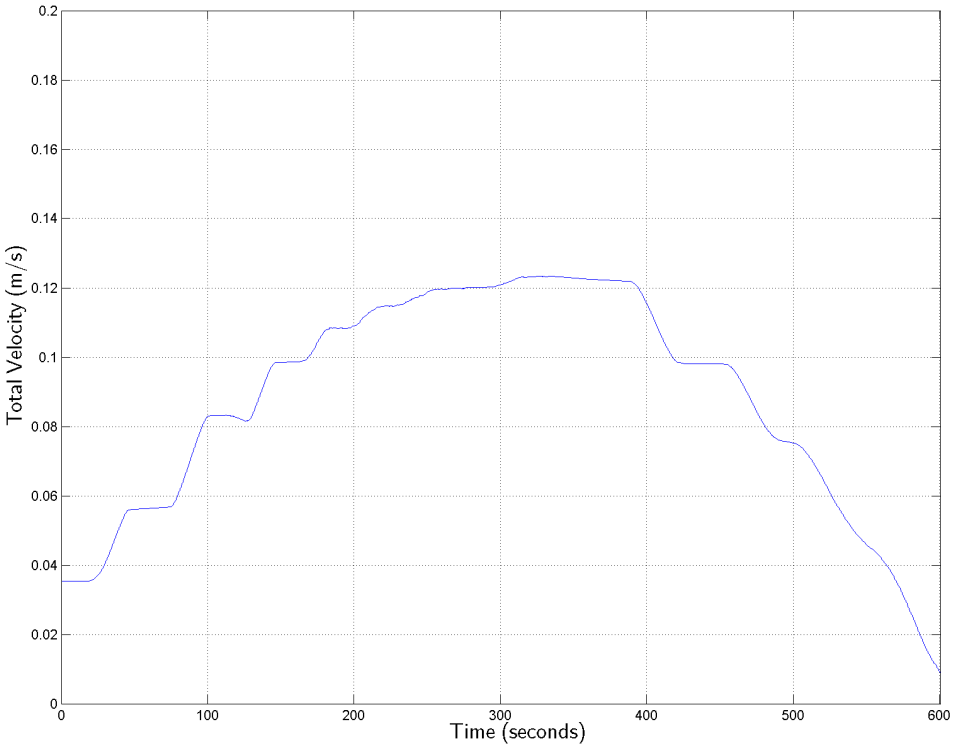

Fig. 97. Relative Velocity Magnitude Profile, Min-Max-Max Inertia Case

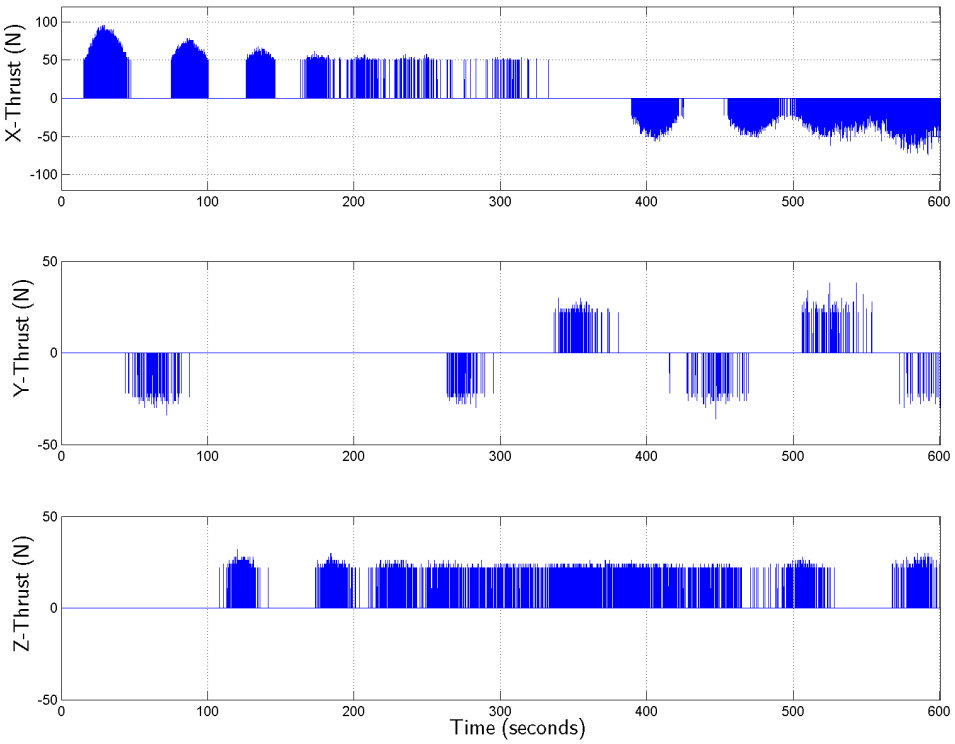

Fig. 98. Chaser Thrust Profile, Min-Max-Max Inertia Case

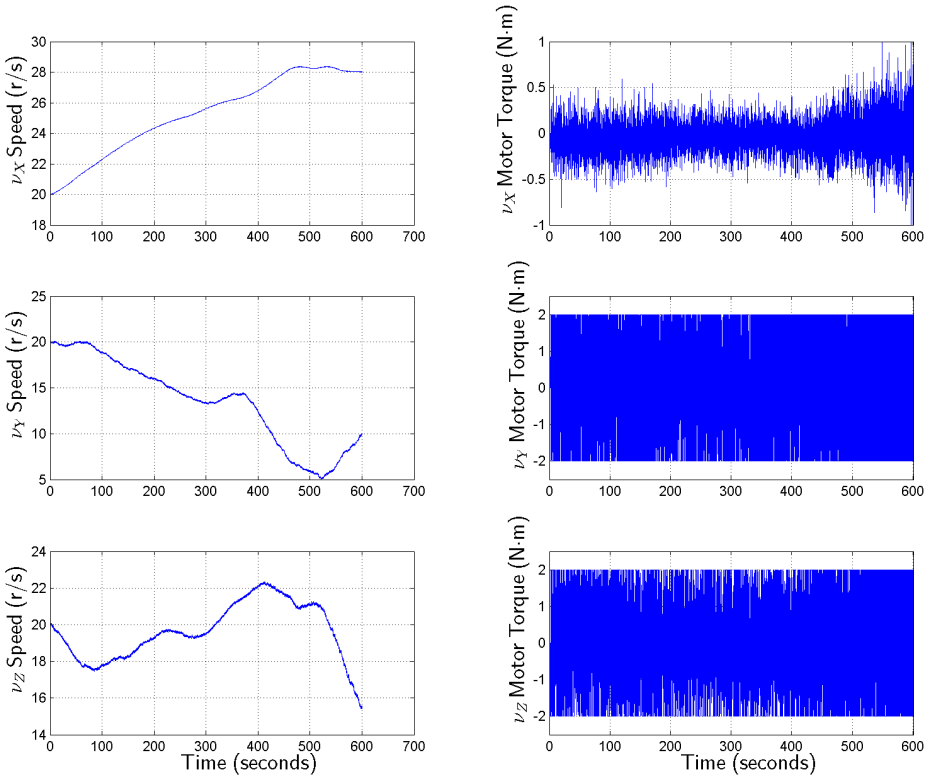

Fig. 99. Wheel Speeds & Motor Torques, Min-Max-Max Inertia Case

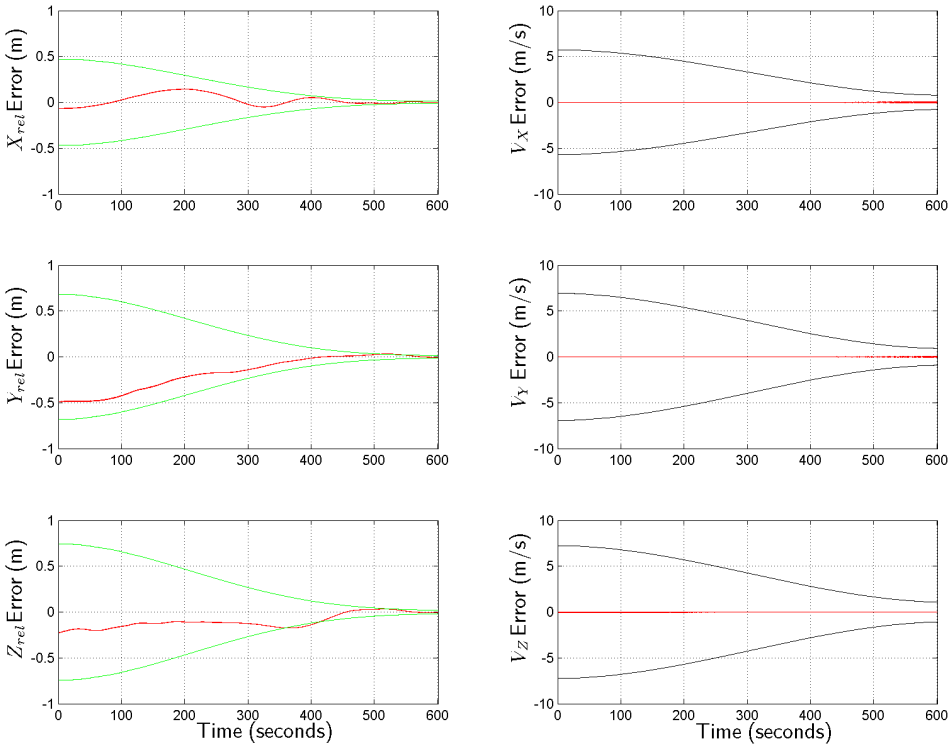

Fig. 100. Position & Velocity Estimate Errors, Min-Max-Max Inertia Case

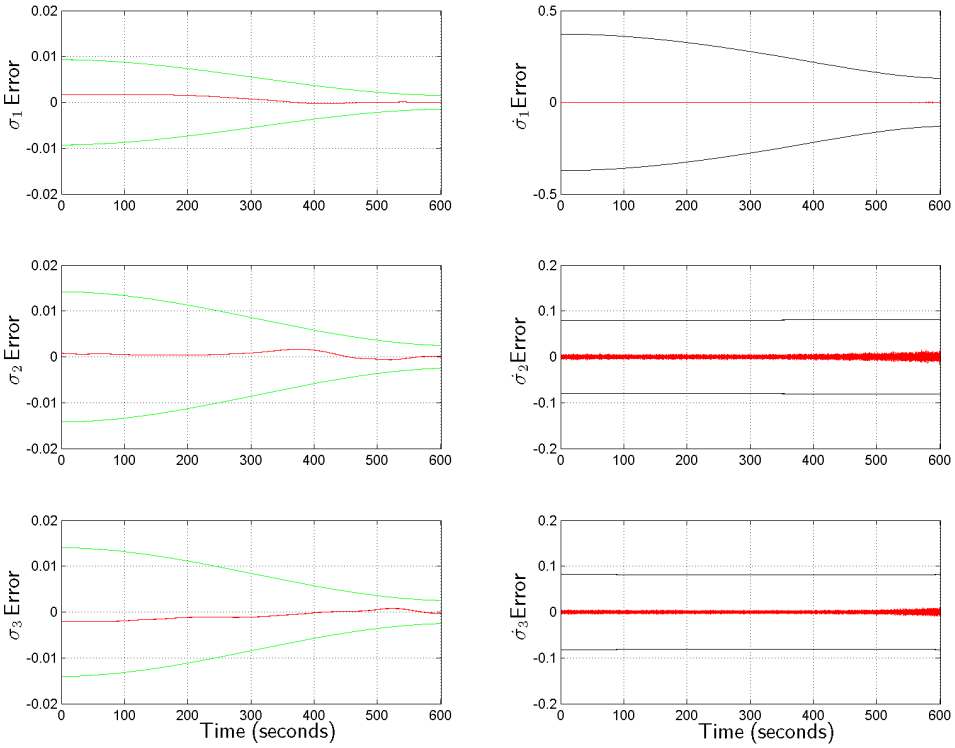

Fig. 101. Orientation & Attitude Rate Estimate Errors, Min-Max-Max Inertia Case

Other results for the current test case are displayed in Figures 98 through 101. For brevity, the plots will not be discussed specifically since they are very similar to those from the nominal case. It will suffice to summarize them by stating that the performance of the controller, VisNav, and the Kalman filter were all excellent throughout this case, despite the statistically highly unlikely worst on worst on worst maximum inertia error scenario it represented.

Further discussion of these results will take place in Sub-section 14, where they will be considered in the context of the controller #2 portion of the experiment as a whole.

8. Test Case 7: Max-Max-Max Initial Attitude Error

Test Case 7 for controller #2 evaluated the performance robustness of the docking controller against maximum *positive* initial relative attitude error in all three axes simultaneously. These values were each a +7.5° deviation from the target vehicle angles in each axis. Note that unlike the previous controller #2 test cases, there was no corresponding controller #1 test on which to base the values for this one on. 7.5° was chosen to stay well within the VisNav sensor's "sweet spot" of field-of-view while still being a significant offset in each axis. All other simulation parameters besides initial relative attitude were held at nominal values for the entire run—including chaser mass, chaser moments of inertia, and initial starting position. Since VisNav and the Kalman filter were used in this test, sensor noise was also present as for every case.

Figure 102 shows the current case's target frame relative trajectory. This plot is a bit more interesting than the trajectory plots for the previous test cases have been, since it is not quite as similar to the nominal case as they were. However, even though there is a bit more deviation from a straight line in this one, it is still

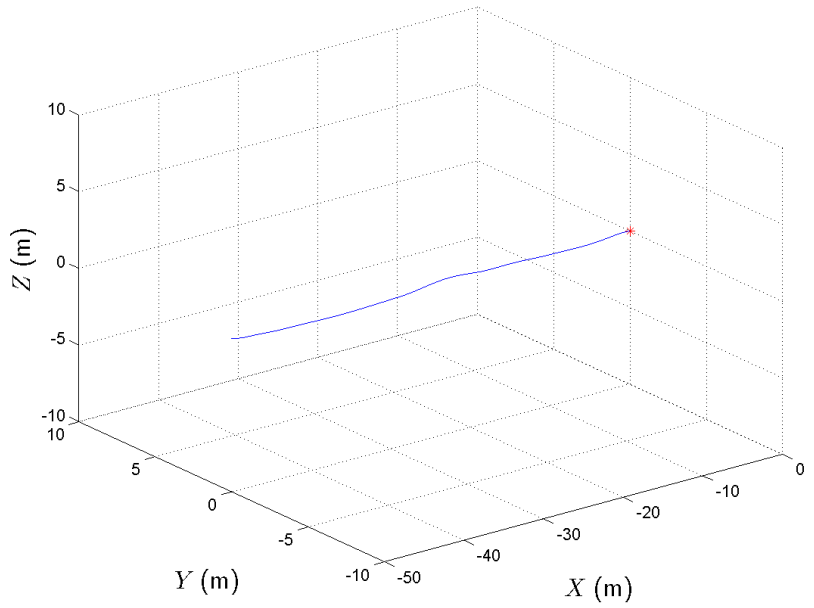

Fig. 102. Chaser Relative Trajectory, Max-Max-Max Attitude Case

apparent that the controller easily managed the given scenario even though it was a statistically unlikely worst on worst on worst case. Since attitude deviation should arguably be the hardest type of error for the docking system to overcome because of its reliance upon a vision-based rel-nav sensor (which necessarily requires a certain level of pointing accuracy to generate a useful solution), the fact that the controller appears to have handled the current case so well is very favorable.

Figures 103 and 104 display the relative states with respect to time for this case. Notice that there is more significant deviation in y and z position during the maneuver than has been seen previously. The angular rates are also larger, indicating that the attitude control system was working harder than in previous cases. However, in spite of these differences from the nominal, the plots still show that the controller had acceptable performance characteristics throughout the run and especially at docking. The final position error for this test case was an excellent 1.33 cm, while the final

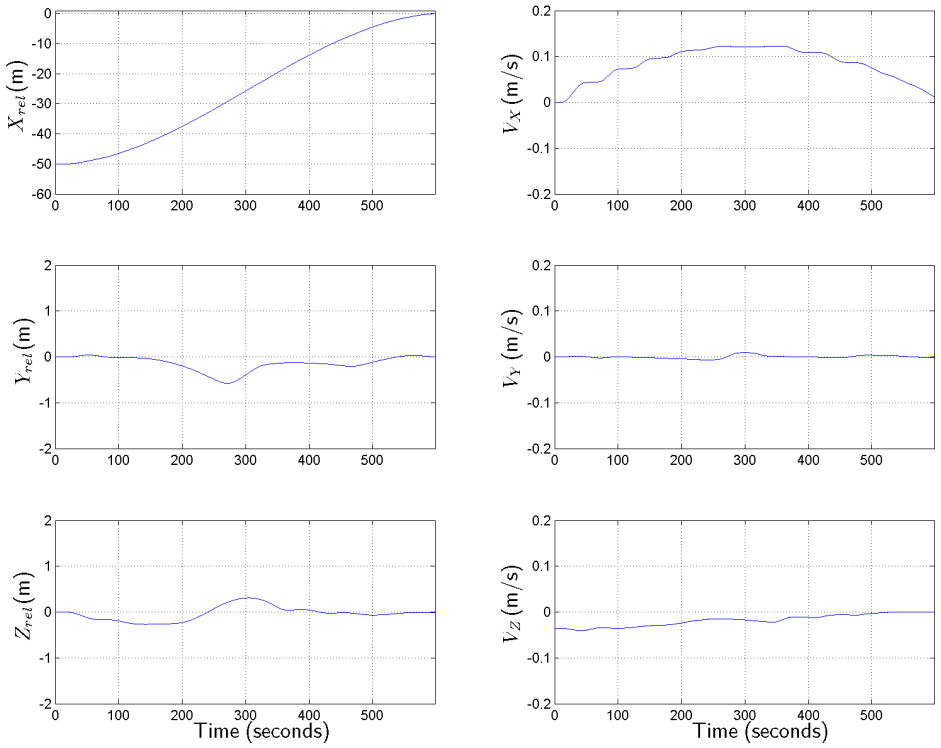

Fig. 103. Chaser Relative Position & Velocity, Max-Max-Max Attitude Case

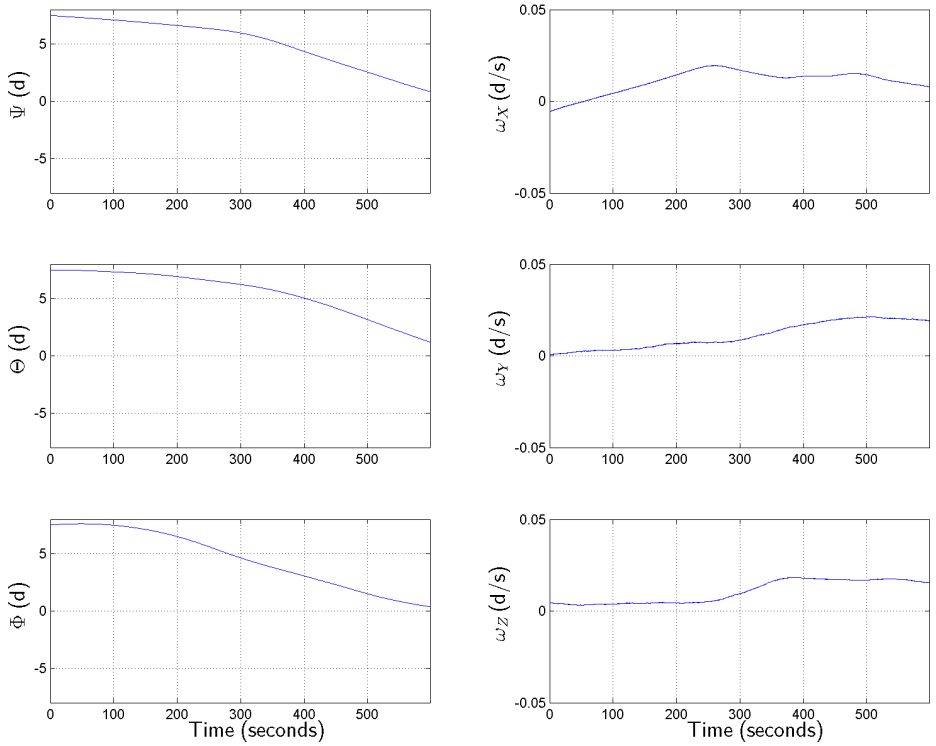

Fig. 104. Chaser Relative Orientation & Attitude Rate, Max-Max-Max Attitude Case

attitude errors were 0.86, 1.21, and 0.38 degrees in yaw, pitch, and roll respectively. Note that this was the first test case so far in which any of the attitude angles failed to achieve the "preferred" criteria at docking—pitch came up a bit short, though it still cleared the "required" criteria by a comfortable amount. The time at docking for this case was 598.38 seconds. So, according to the proscribed success criteria, the final values for position, attitude, and time were all good enough for this to be considered a successful dock.

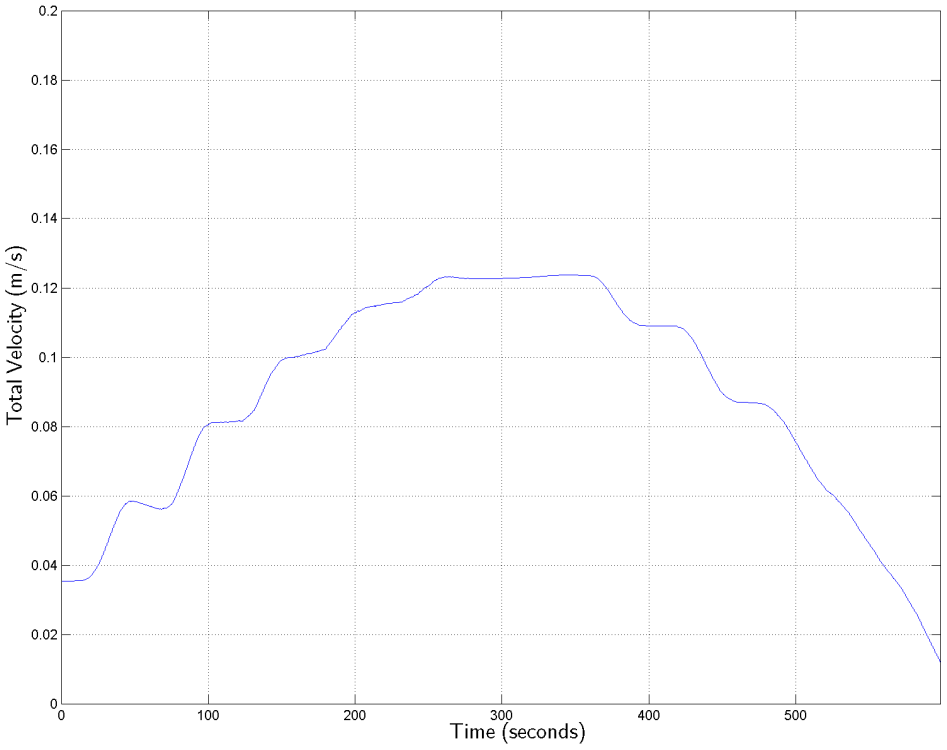

Fig. 105. Relative Velocity Magnitude Profile, Max-Max-Max Attitude Case

The total relative velocity magnitude versus time plot for Test Case 7 is shown in Figure 105. Inspection of the figure reveals that the total velocity behaves desirably for the entire test run, including at docking. The final total velocity for this case

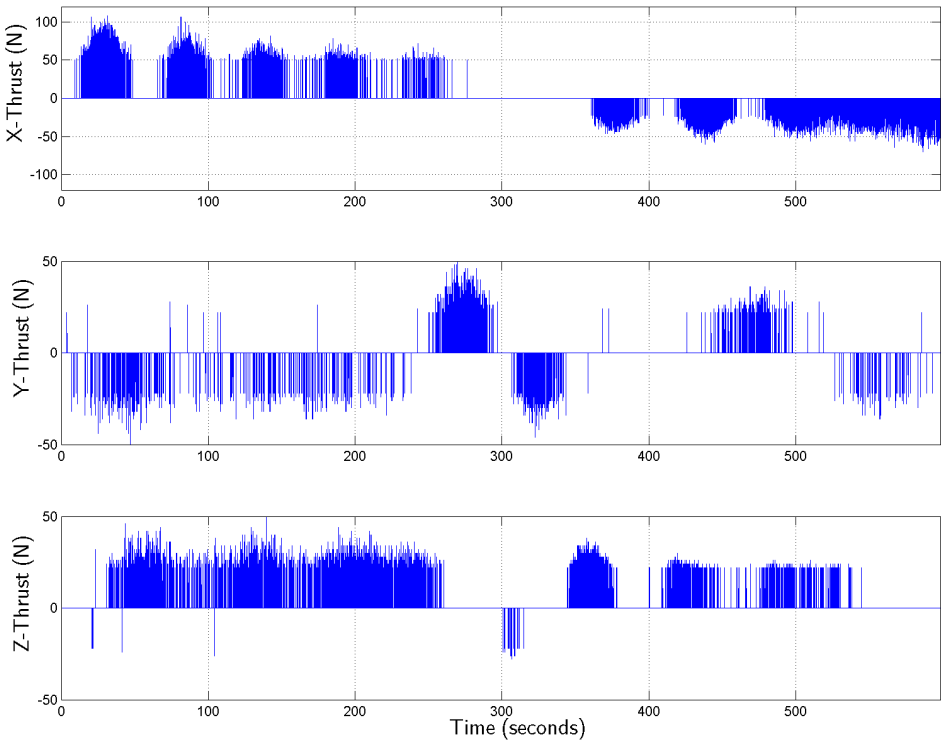

Fig. 106. Chaser Thrust Profile, Max-Max-Max Attitude Case

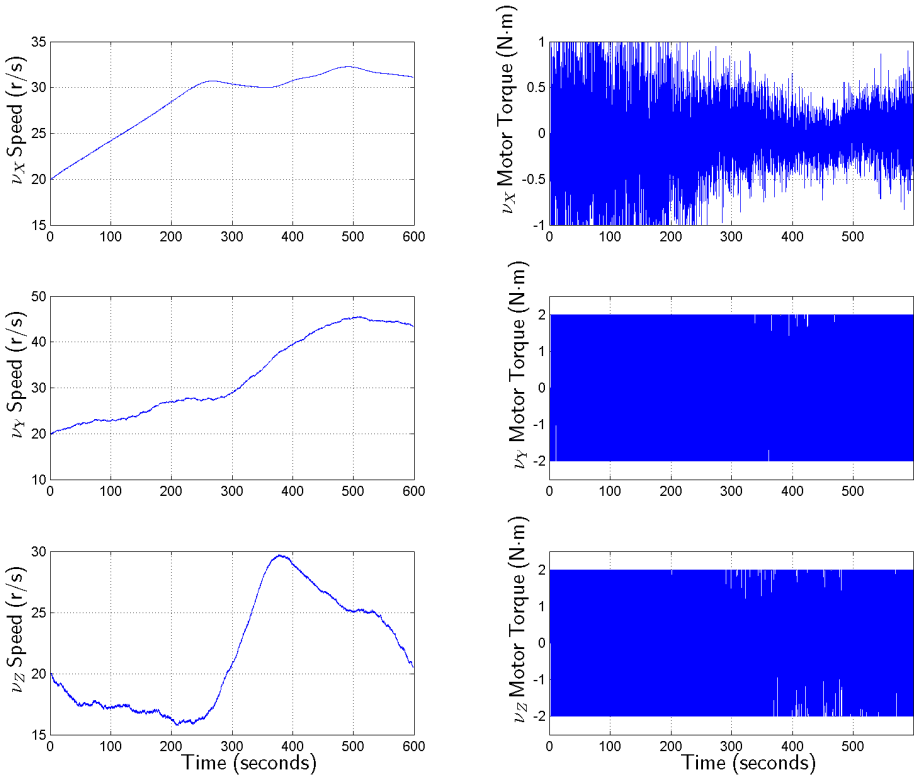

Fig. 107. Wheel Speeds & Motor Torques, Max-Max-Max Attitude Case

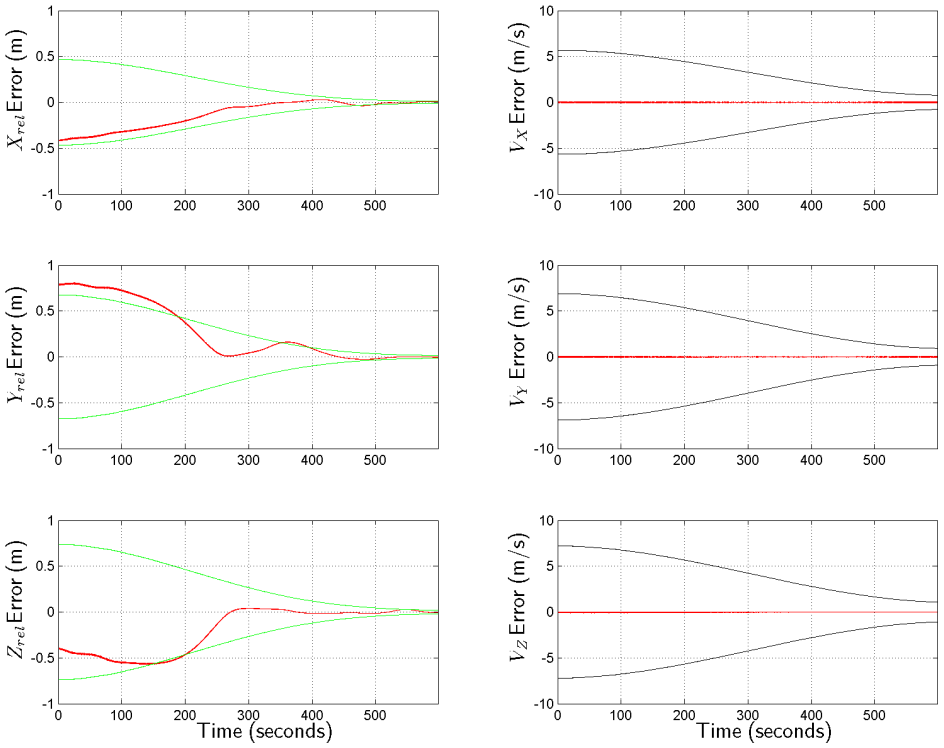

Fig. 108. Position & Velocity Estimate Errors, Max-Max-Max Attitude Case

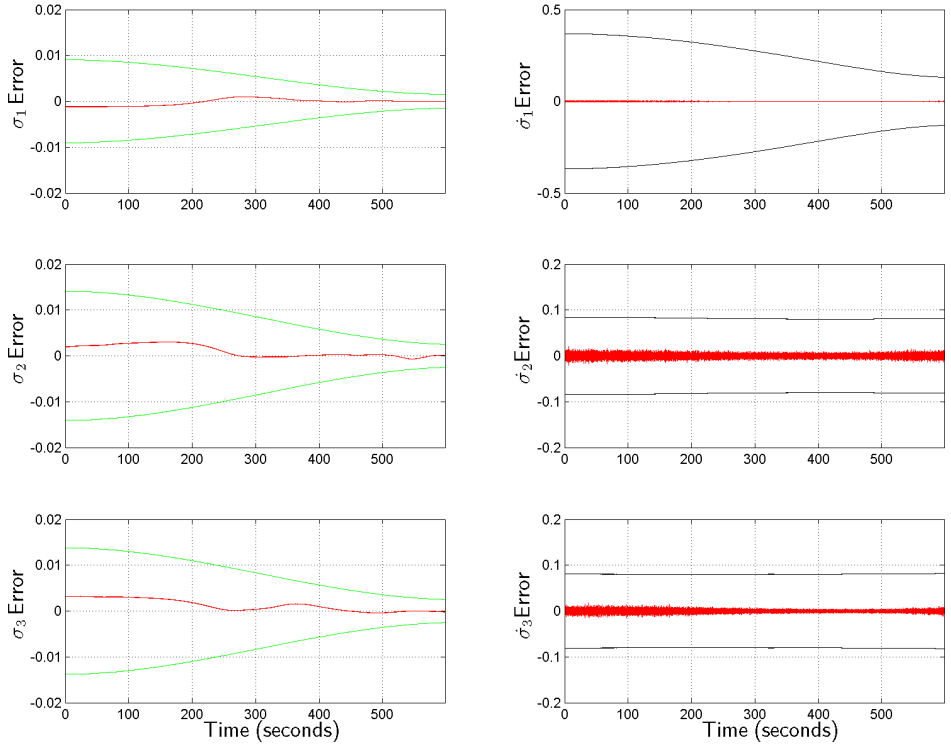

Fig. 109. Orientation & Attitude Rate Estimate Errors, Max-Max-Max Attitude Case

was 1.21 centimeters per second, which exceeds even the 5 cm/s "preferred" docking velocity criteria quite easily. Thus, but for pitch, this test case could be considered an "exceptional" test run; but it is still a successful dock even if it is not "exceptional" per-se.

The other results for this "max-max-max attitude" test case are reported in Figures 106 through 109. While there is clearly more activity by the z-axis thrusters and all three reaction wheels than for the nominal case, overall the general characteristics of the remaining plots are still pretty similar to the nominal. One should note, however, that the y-position error component does exceed its 3-σ bounds for a good portion of the beginning of the run, though it converges back in as the attitude error declines. This is behavior is not too surprising given some knowledge of the VisNav sensor, since y and yaw are very closely related within it. Thus, at the beginning when the sensor solution is rather poorly conditioned due to the highly deviated attitude state, VisNav struggled to properly resolve y. Then, as the control system began to bring the attitude under control, the solution became better conditioned and the y estimate improved to the levels seen in the nominal case. So, overall, the controller, VisNav, and the Kalman filter each still basically performed very well despite a challenging and statistically highly unlikely worst on worst on worst maximum attitude error scenario.

Macroscopic analysis of this test case's results in the context of the entire experiment can be found in Sub-section 14.

9. Test Case 8: Min-Min-Min Initial Attitude Error

The eighth test case for controller #2 evaluated its performance robustness against maximum *negative* initial relative attitude error in all three axes simultaneously. The values used in this case were a per-axis $-7.5°$ deviation from the target vehicle

orientation; see the previous sub-section for the reasoning behind the choice of 7.5° for the magnitude. The other simulation parameters (besides initial relative attitude) were kept at their nominal values for the whole test case—including chaser mass, chaser moments of inertia, and initial starting position. Since the VisNav sensor and Kalman filter were used, sensor noise was also present in this case.

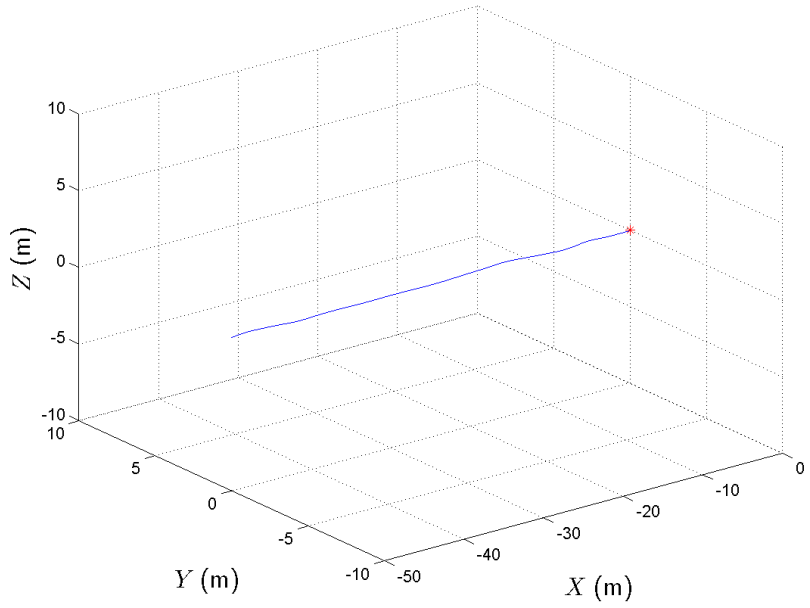

Fig. 110. Chaser Relative Trajectory, Min-Min-Min Attitude Case

The current case's relative trajectory in the target frame is plotted in Figure 110. Interestingly, this plot is more similar to the nominal case plot than it is the max-max-max attitude case plot. This makes controller #2 seem a bit asymmetric on handling attitude errors—it appears it can better acommodate negative attitude angles than positive ones, even of the same magnitude. Aside from that, it is obvious from the figure that the controller easily surmounted this statistically unlikely worst on worst on worst case as well. Since attitude deviation should arguably be the hardest type of error for the docking system to overcome because of its reliance upon a vision-

based rel-nav sensor (which necessarily requires a certain level of pointing accuracy to generate a useful solution), this result too is very favorable.

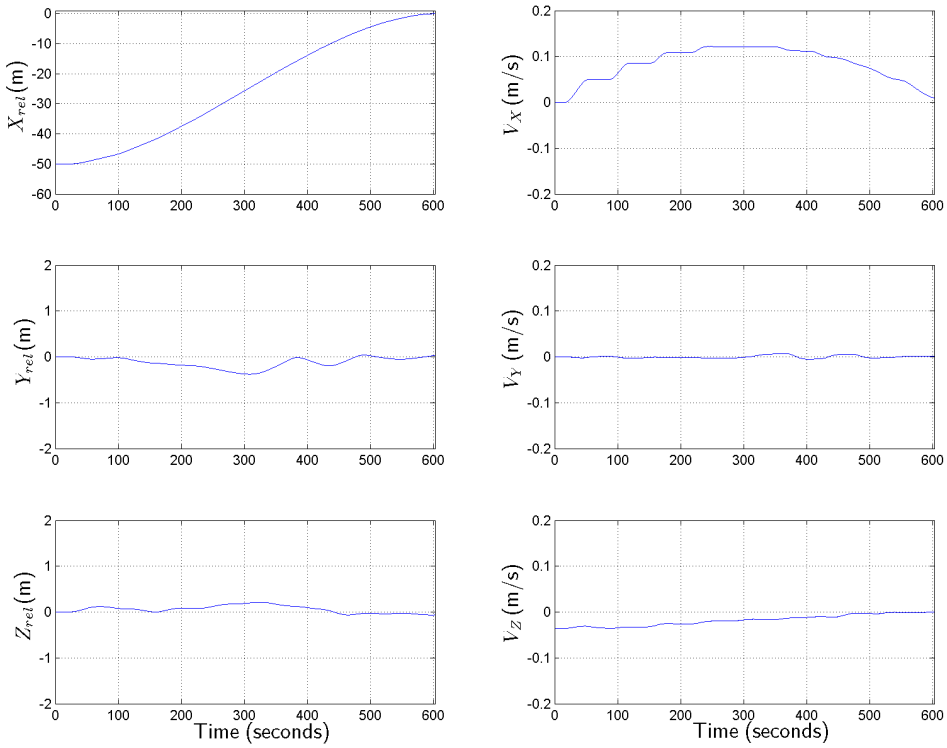

Fig. 111. Chaser Relative Position & Velocity, Min-Min-Min Attitude Case

Figures 111 and 112 show the time histories of the current case's chaser relative states. These also generally more closely resemble the nominal case than the max-max-max attitude case, except for the angular velocities which favor (but with opposite sense as one would expect) the max-max-max behavior. Overall, the characteristics of the run that can be gleaned from these plots reveal it to be very satisfactory. The total position error at docking for the current case was 6.84 cm, while the final attitude errors were -0.97, -0.74, and -0.33 degrees in yaw, pitch, and roll respectively. The final docking time for this case was 602.44 seconds. Thus, the final

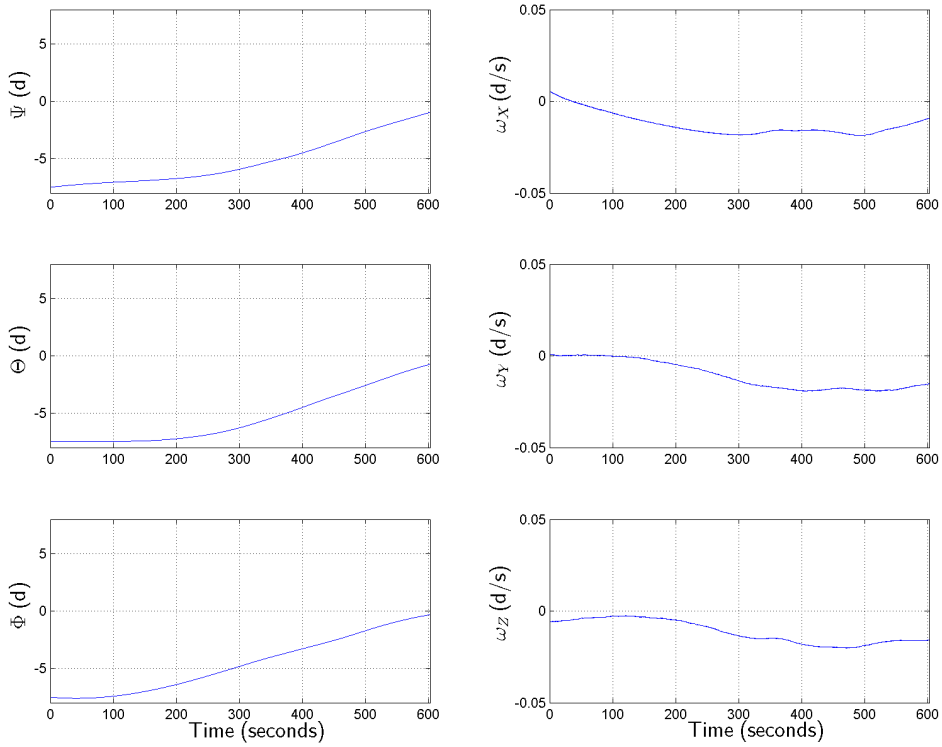

Fig. 112. Chaser Relative Orientation & Attitude Rate, Min-Min-Min Attitude Case

values for position, attitude, and time were all good enough for this to be considered a successful dock, with the final attitude errors additionally exceeding their respective preferred criteria.

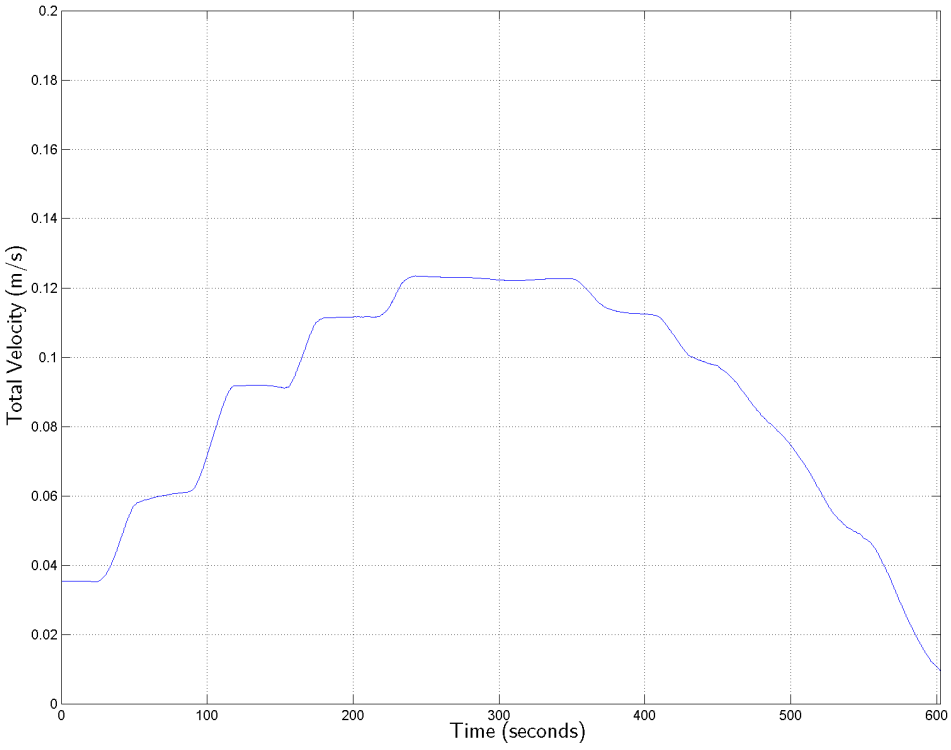

Fig. 113. Relative Velocity Magnitude Profile, Min-Min-Min Attitude Case

The total relative velocity magnitude time history for Test Case 8 is plotted in Figure 113. The figure shows that the total velocity behaves desirably for the entire test run, especially during the second half of the trajectory. The current case's final total velocity was 0.95 centimeters per second, which easily surpasses the 5 cm/s "preferred" docking velocity criteria. So, this test case qualifies as an "exceptional" test run based on its total velocity and attitude errors at docking.

The remaining results for the current "min-min-min attitude" test case are shown

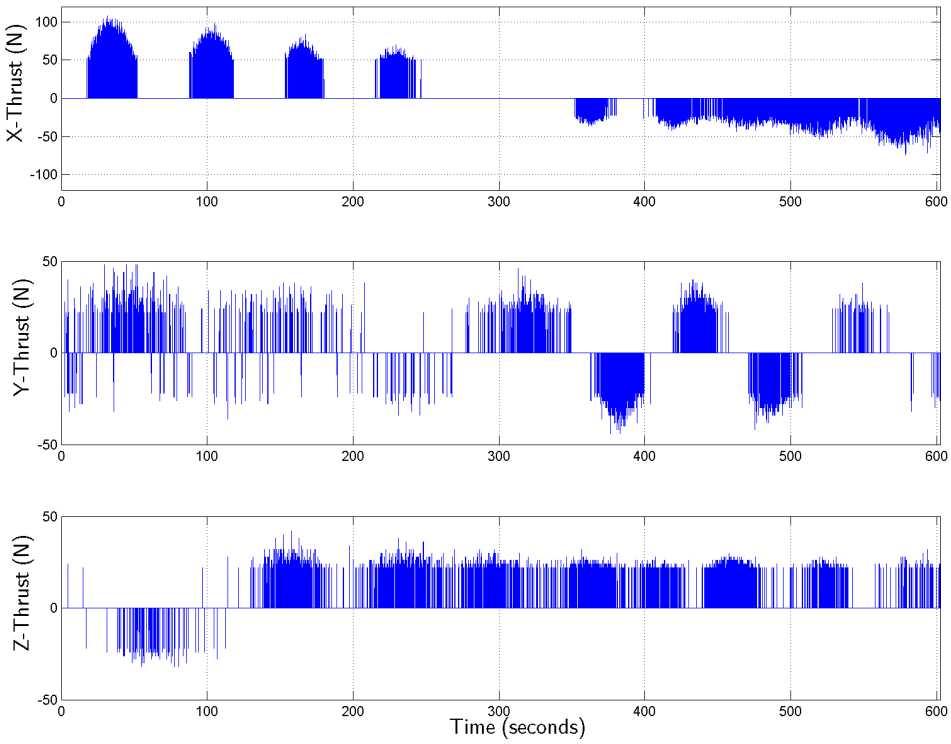

Fig. 114. Chaser Thrust Profile, Min-Min-Min Attitude Case

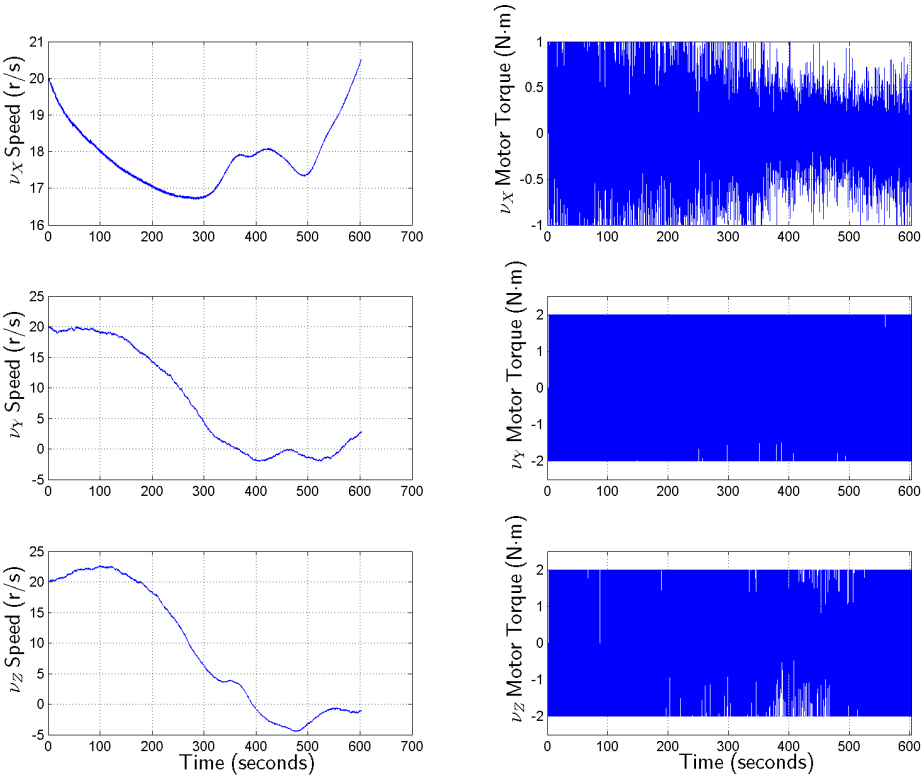

Fig. 115. Wheel Speeds & Motor Torques, Min-Min-Min Attitude Case

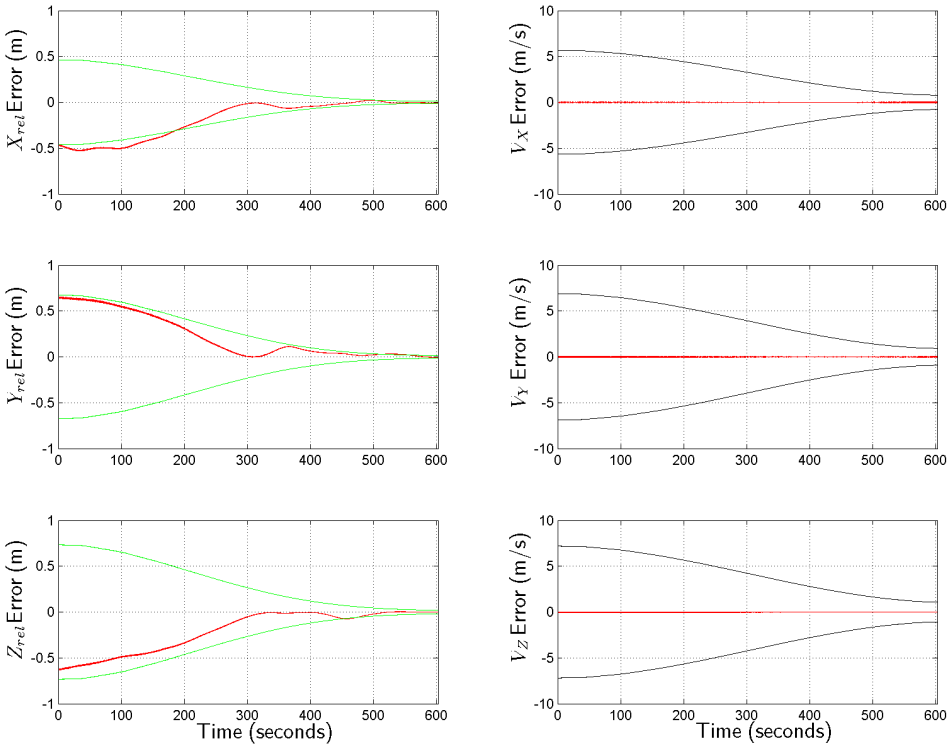

Fig. 116. Position & Velocity Estimate Errors, Min-Min-Min Attitude Case

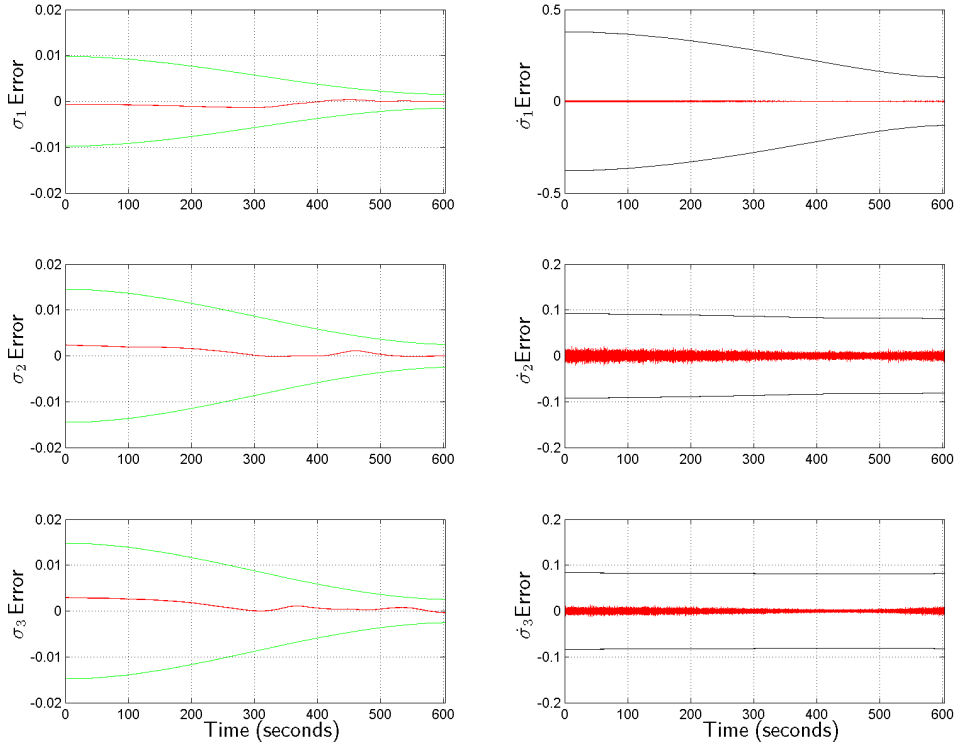

Fig. 117. Orientation & Attitude Rate Estimate Errors, Min-Min-Min Attitude Case

in Figures 114 through 117. While there is clearly more activity by the z-axis thrusters and all three reaction wheels than for the nominal case, the general characteristics of the other plots are again more similar to the nominal than the max-max-max attitude case. One unique thing of note is that the x-position error component, rather than y in the previous case, slightly exceeds its 3-σ bounds for the beginning of the run, though it converges back in as the attitude error declines. This is a somewhat surprising result that probably bears some further investigation; while the y error in the previous case is explained by the previously mentioned state coupling within the VisNav sensor, there is not an immediately obvious reason for why x would have deviated while y did not. Besides this slight deviation in x, the controller, VisNav, and the Kalman filter each otherwise basically performed very well in a challenging and statistically highly unlikely worst on worst on worst maximum attitude error test case.

Further analysis of this test case's results in the context of the entire experiment will be conducted in Sub-section 14.

10. Test Case 9: Max-Min-Min Initial Attitude Error

Controller #2 Test Case 9 evaluated the controller's performance robustness against maximum positive initial relative attitude error in yaw, simultaneous with maximum negative initial relative attitude error in both pitch and roll. In all axes, this resulted in a 7.5° deviation from the target vehicle orientation in the appropriate direction. (Sub-section 8 elaborates on the reasoning behind the choice of 7.5° for the angles.) The other simulation parameters aside from chaser's initial relative attitude were help nominal throughout—including chaser mass, chaser moments of inertia, and initial starting position. Sensor noise was also present by virtue of the VisNav sensor and Kalman filter, as for all cases.

Figure 118 shows the current case's target frame relative trajectory. At first

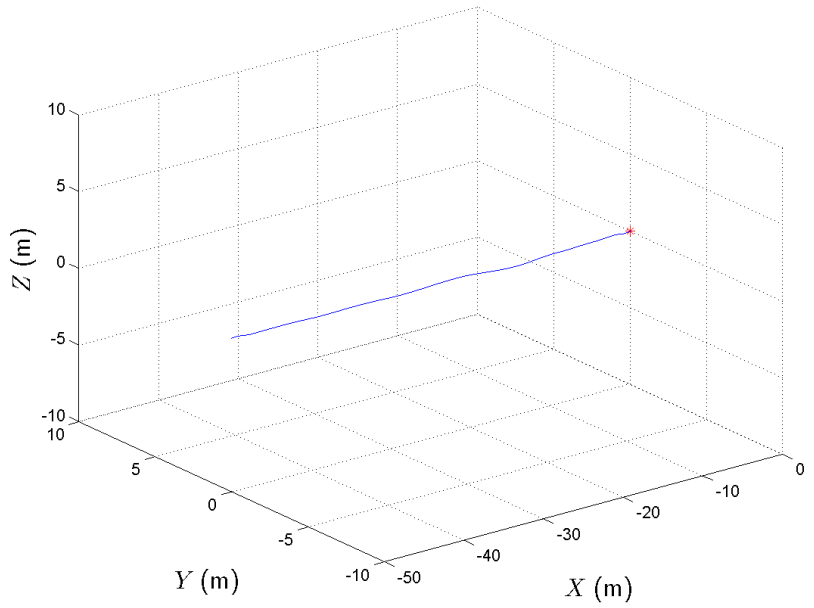

Fig. 118. Chaser Relative Trajectory, Max-Min-Min Attitude Case

glance, the plot strongly resembles the nominal one. However, inspection of the time history plots for the chaser relative states (shown in Figures 119 and 120) reveal some oscillation in y and z position that the controller does not adequately damp out in time for docking. Also, yaw and pitch do not converge as well as in other cases. Thus while this case will qualify as a successful dock "by the numbers", qualitatively the controller's performance leaves something to be desired.

The total position error at docking was 7.05 cm, with final attitude errors of 0.75, -0.96, and -0.24 degrees respectively in yaw, pitch, and roll. The docking maneuver completed for this case at 601.07 seconds. These final values for position, attitude, and time all exceeded their respective criteria for a successful dock, with the final attitude errors also (barely for yaw and pitch) exceeding their assigned "preferred" criteria.

The Test Case 9 total relative velocity magnitude time history is graphed in

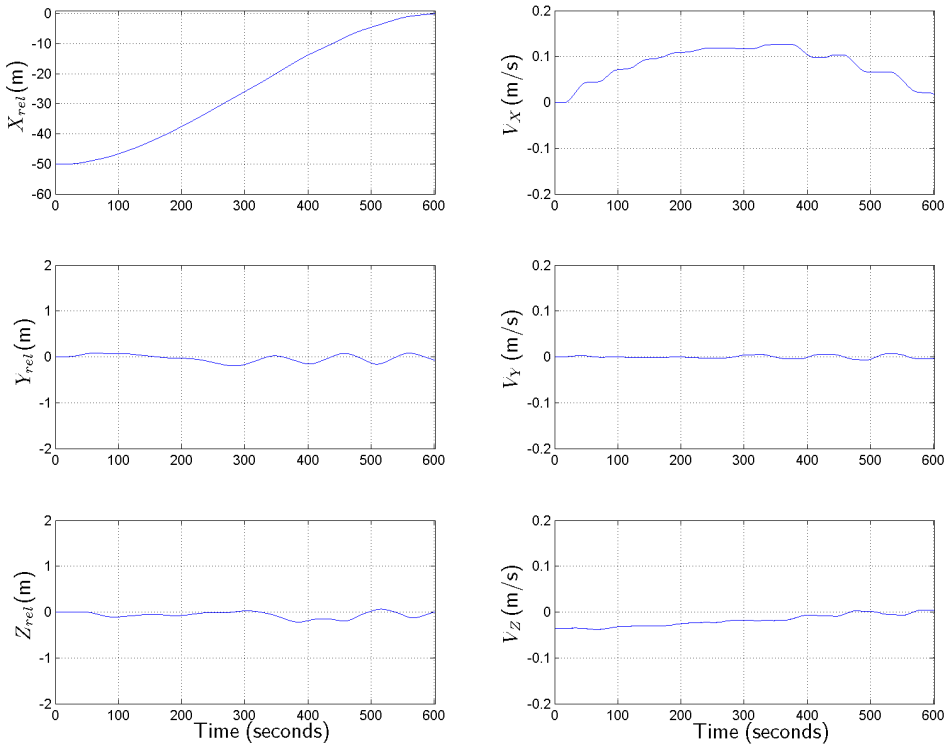

Fig. 119. Chaser Relative Position & Velocity, Max-Min-Min Attitude Case

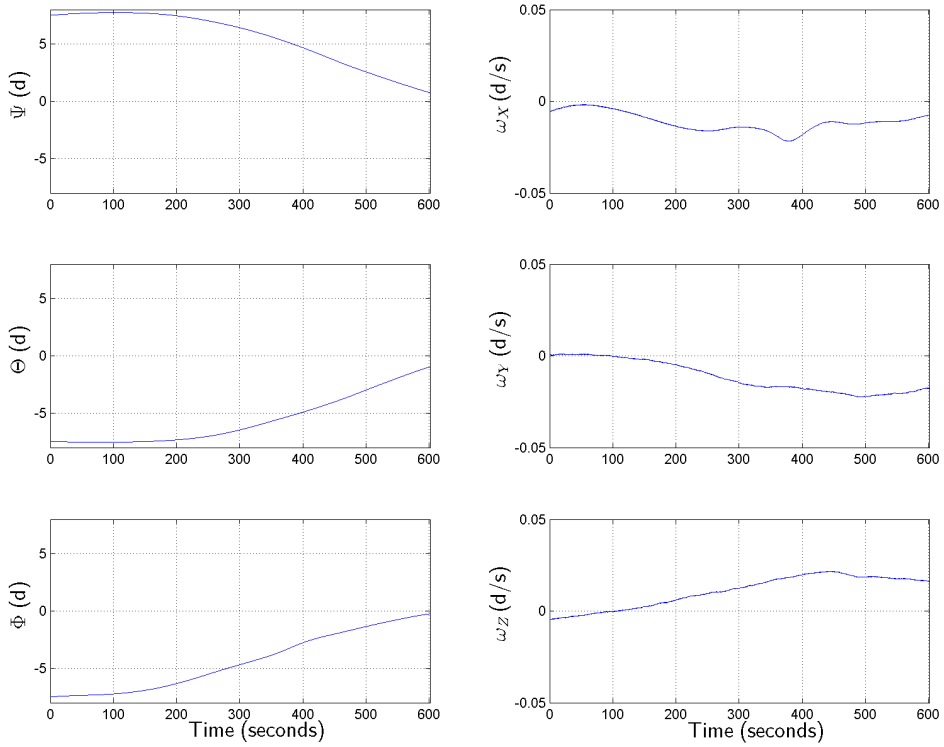

Fig. 120. Chaser Relative Orientation & Attitude Rate, Max-Min-Min Attitude Case

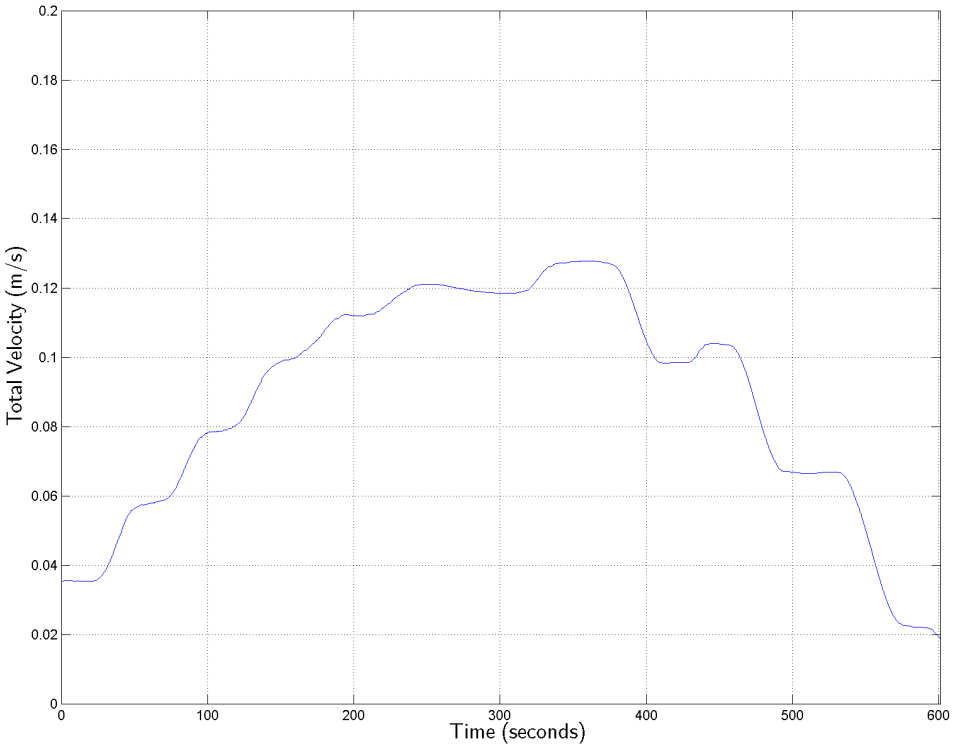

Fig. 121. Relative Velocity Magnitude Profile, Max-Min-Min Attitude Case

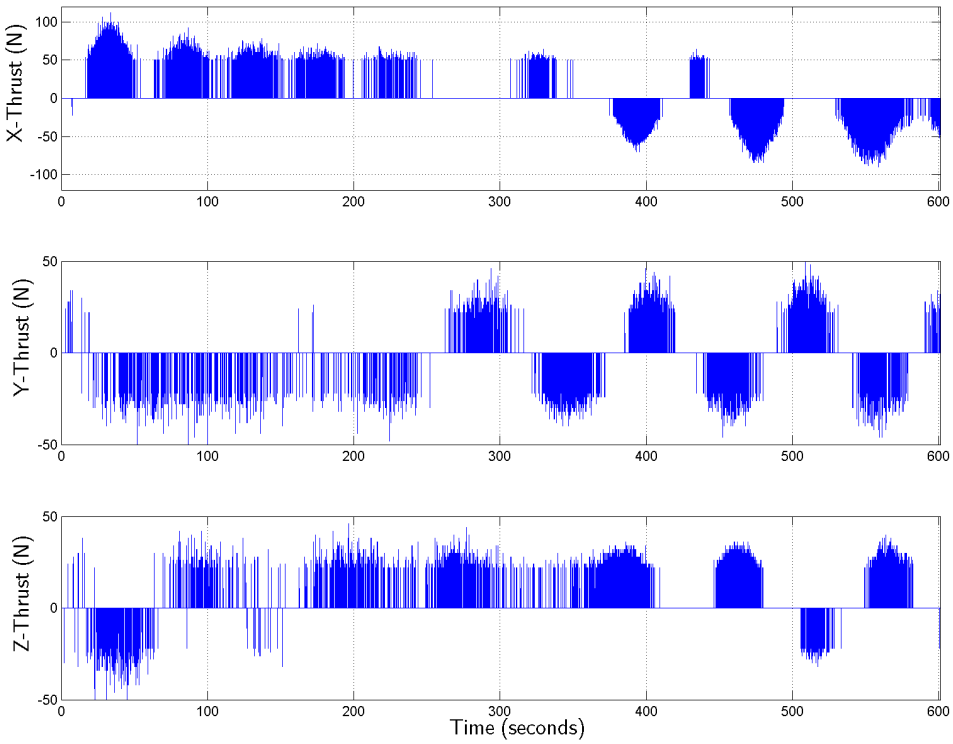

Fig. 122. Chaser Thrust Profile, Max-Min-Min Attitude Case

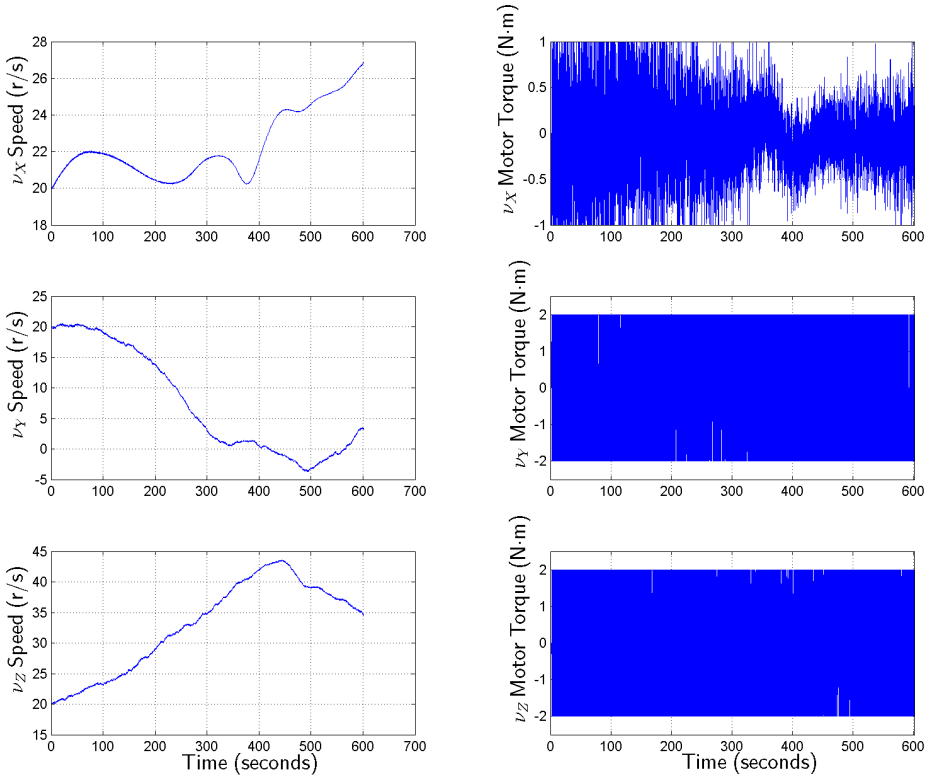

Fig. 123. Wheel Speeds & Motor Torques, Max-Min-Min Attitude Case

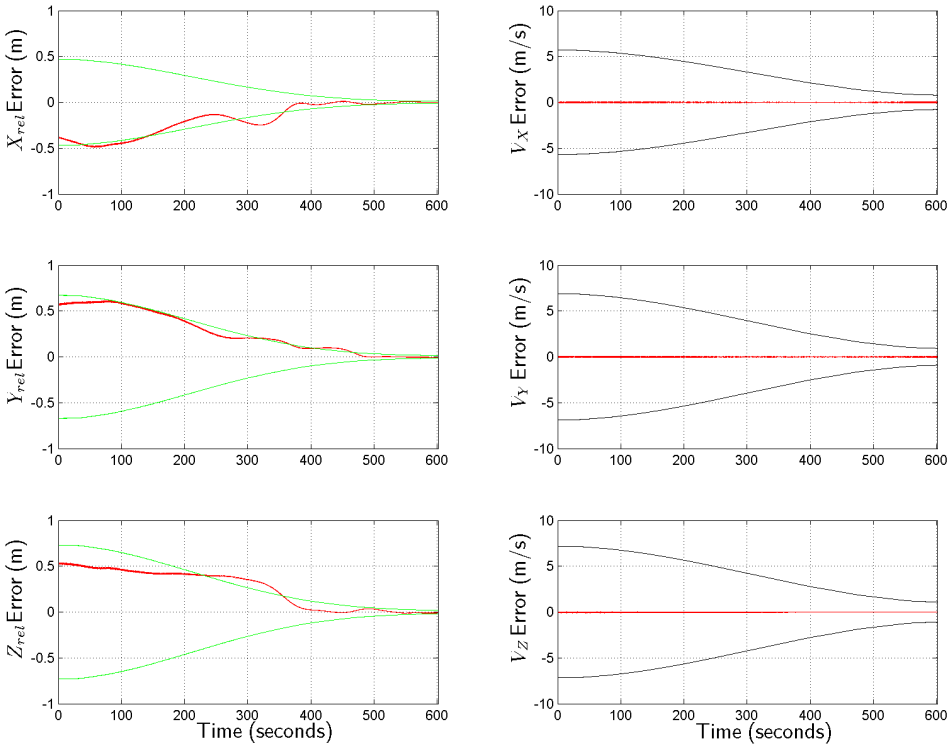

Fig. 124. Position & Velocity Estimate Errors, Max-Min-Min Attitude Case

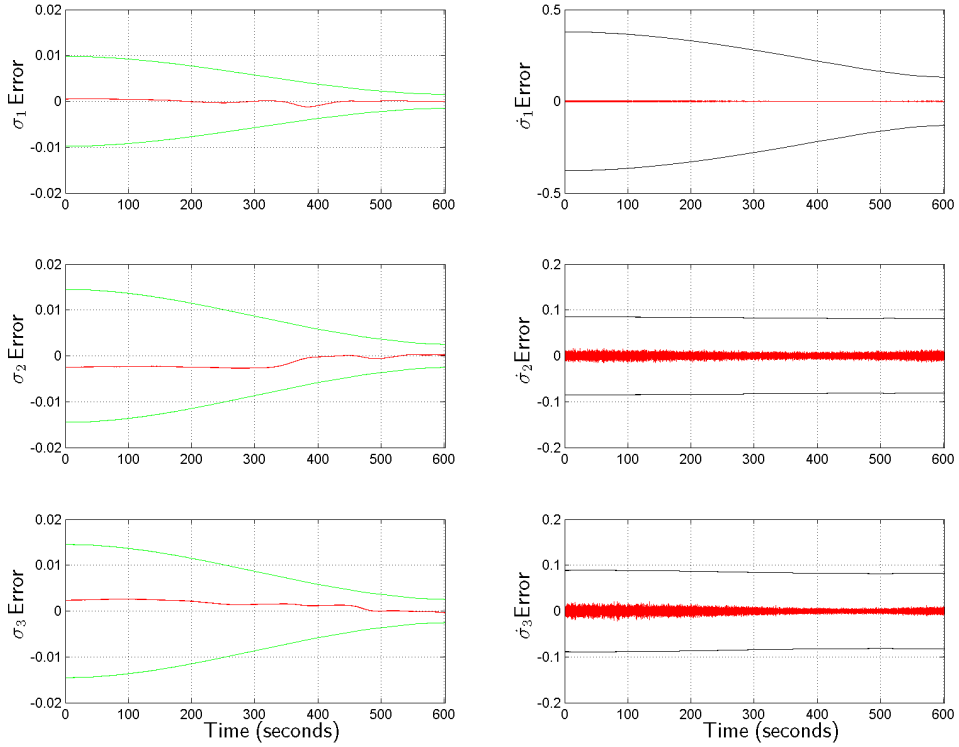

Fig. 125. Orientation & Attitude Rate Estimate Errors, Max-Min-Min Attitude Case

Figure 121. The total velocity appears to behave favorably for the duration of the run, though it exhibits a characteristic "wobble" due to the simultaneous oscillations in y and z. The final total velocity at docking was 1.88 centimeters per second, easily eclipsing the "preferred" docking velocity criteria of 5 cm/s. So, this test case qualifies as an "exceptional" one based strictly on its total velocity and attitude errors at docking. The other plots for this "max-min-min attitude" case, shown in Figures 122 through 125, are consistent with the ones already examined here, and thus will not be specifically discussed.

Since the controller met all docking success criteria but did not qualitatively perform as well as would be desired, further investigations were performed to determine the root cause, as well as to attempt to improve upon its behavior in this case.

Studying the results just presented more closely, it became apparent that the inability of the controller to completely damp out the oscillations in y and z position before docking was due to limits on its fine control in position (which is actually a function of the propulsion system in this case, as will be addressed in a moment). Initial conditions in all of the other cases were such that these limits were not encountered, but this particular combination of max yaw and minimum pitch and roll proved to need more fine control. Similarly, the incomplete convergence of yaw and pitch appeared due to saturation limits in the reaction wheels limiting the controller's attitude control authority. While these limits were encountered frequently in the other cases (review the motor torque plots for any of the other ones to see this), the limits did not become significant for the others because the controller was still able to "get there". However, for this one it could not because of the particular characteristics of the given combination of initial attitude values.

To evaluate these observations, a re-run of this test case was conducted with the 5% deadband removed from the thruster system (to increase the available position fine

control), and the saturation limits removed from the reaction wheel drive motors (to increase attitude control authority). This proved that the observations were correct: the position transients in y and z were adequately damped out before docking, and the attitude convergence was improved for yaw and pitch. However, the attitude still did not completely converge. So, finally, the attitude gains in the controller were increased as well, and the controller system's performance then became fully satisfactory. Figures 126 through 133 summarize the improved results for this test run, which shows that these issues were in fact the source of the less than desirable performance by the baseline controller #2 in this case.

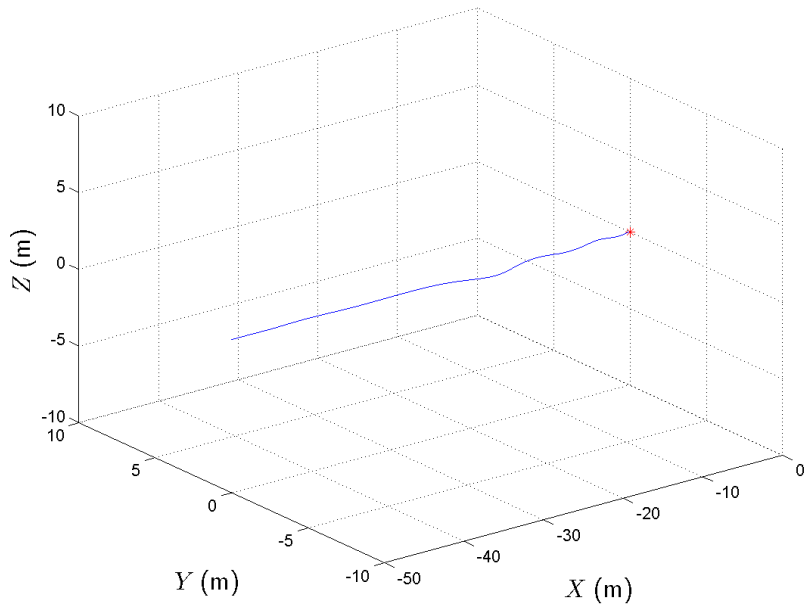

Fig. 126. Chaser Relative Trajectory, Redone Max-Min-Min Attitude Case

The total position error at docking for this modified case was 2.58 cm, an improvement over the baseline for this case. Final attitude errors were 0.53, -0.54, and -0.25 degrees respectively in yaw, pitch, and roll, which also were improved as compared to the baseline. The docking maneuver completed for this case at 603.45

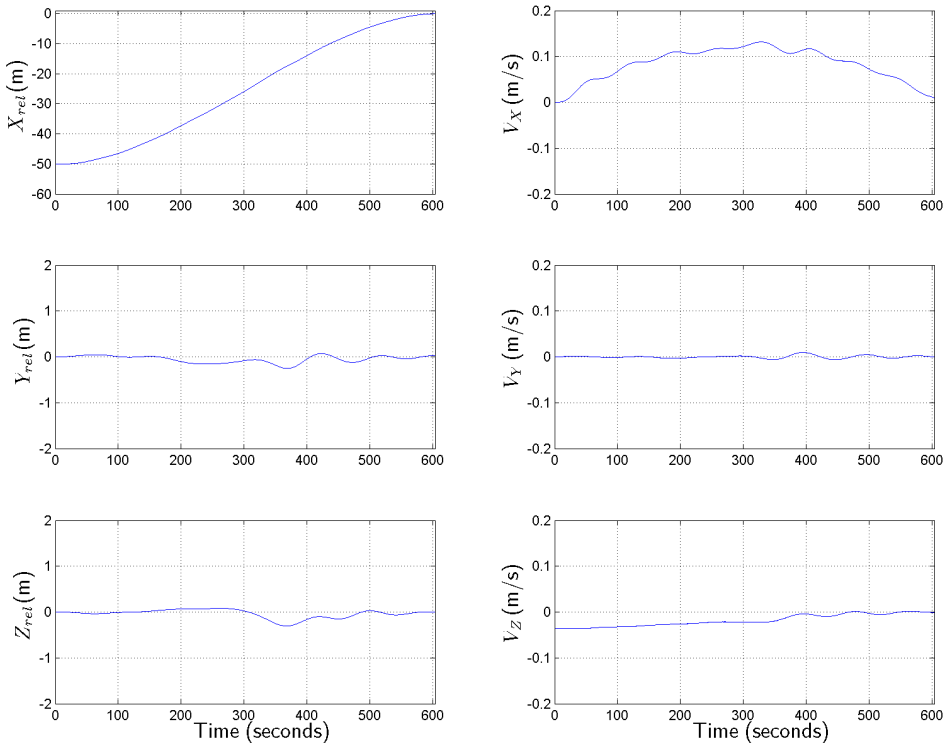

Fig. 127. Chaser Relative Position & Velocity, Redone Max-Min-Min Attitude Case

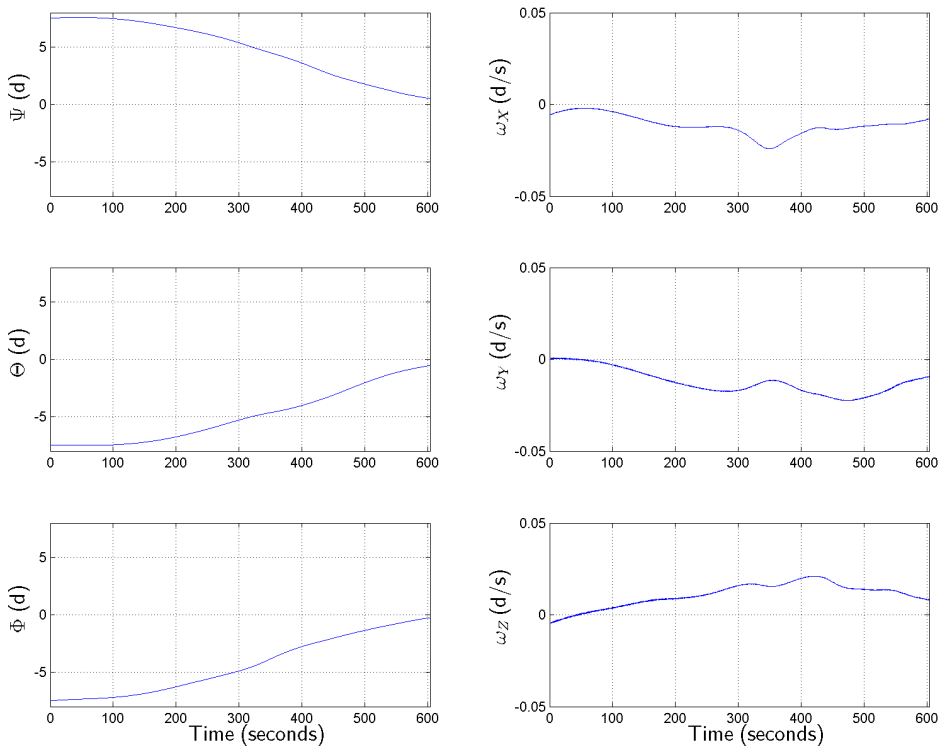

Fig. 128. Chaser Relative Orientation & Attitude Rate, Redone Max-Min-Min Attitude Case

seconds, and the final total velocity at docking was 1.03 centimeters per second (an improvement over the baseline as well). These final values all exceeded their respective criteria for a successful dock, with the final attitude errors and total final velocity all exceeding their assigned "preferred" criteria.

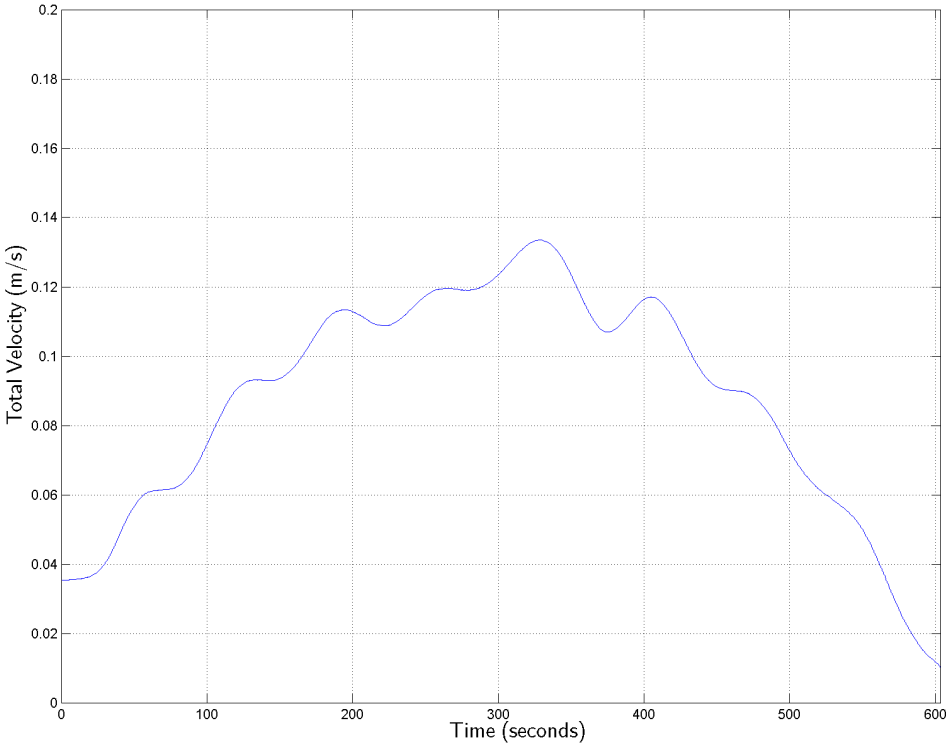

Fig. 129. Relative Velocity Magnitude Profile, Redone Max-Min-Min Attitude Case

The results of this test case will be further discussed in the context of the entire experiment in Sub-section 14.

11. Test Case 10: Min-Max-Max Initial Attitude Error

Test Case 10 for controller #2 evaluated its performance robustness against maximum *negative* initial relative attitude error in yaw, as well as with maximum *positive* initial

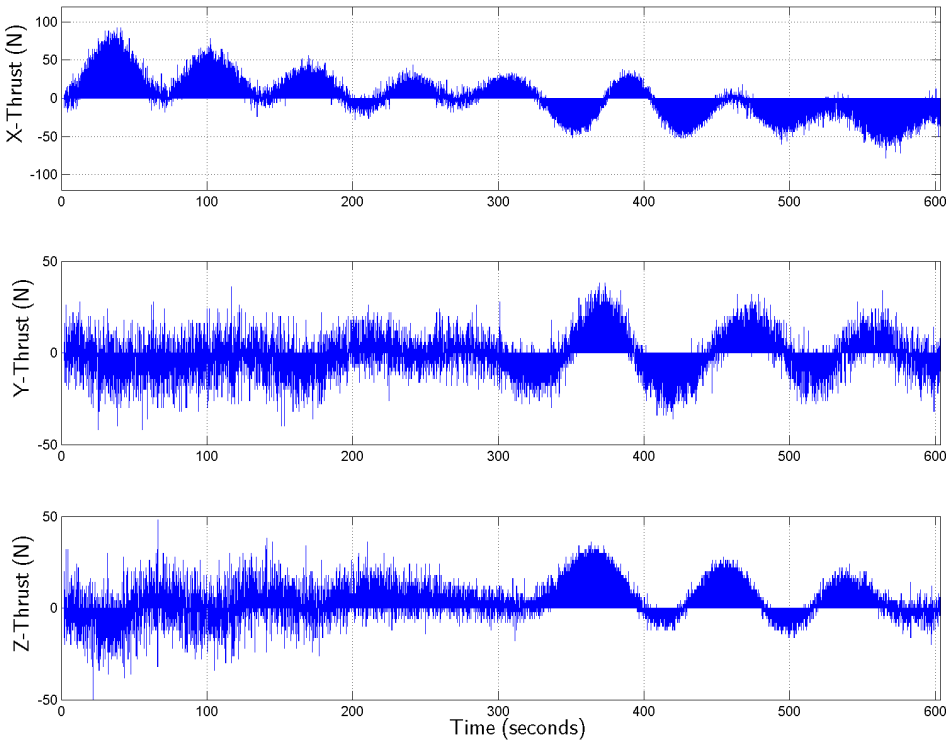

Fig. 130. Chaser Thrust Profile, Redone Max-Min-Min Attitude Case

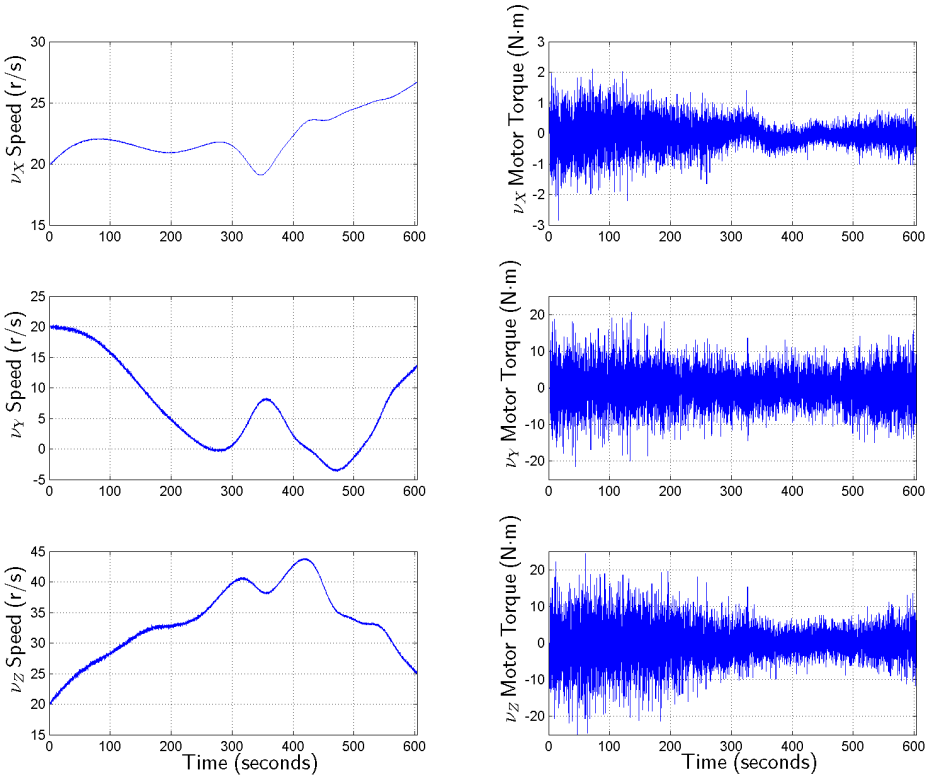

Fig. 131. Wheel Speeds & Motor Torques, Redone Max-Min-Min Attitude Case

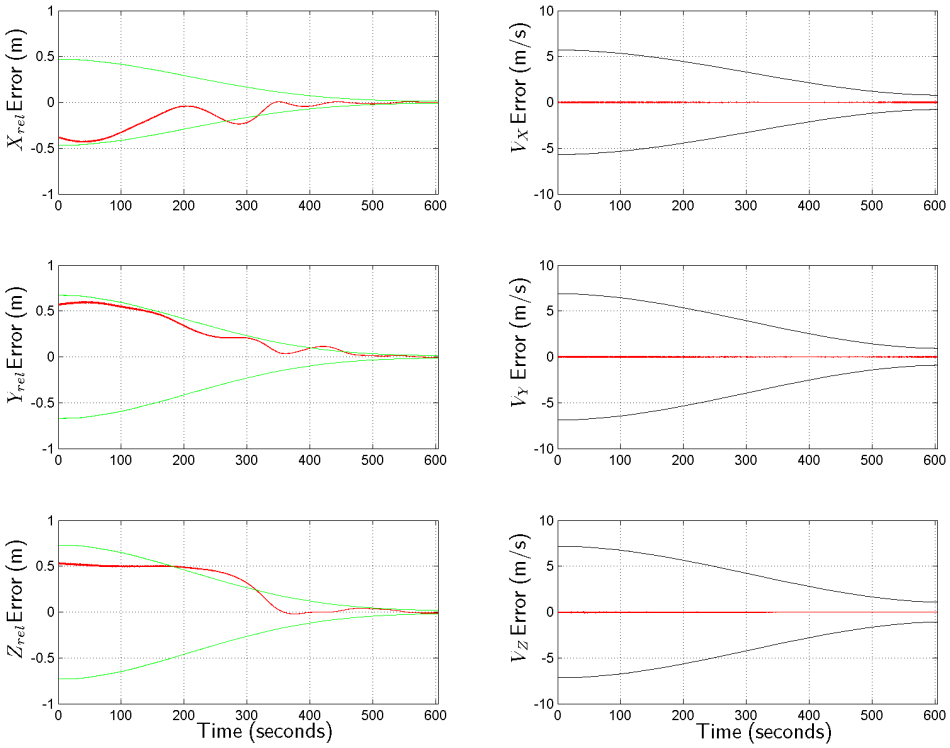

Fig. 132. Position & Velocity Estimate Errors, Redone Max-Min-Min Attitude Case

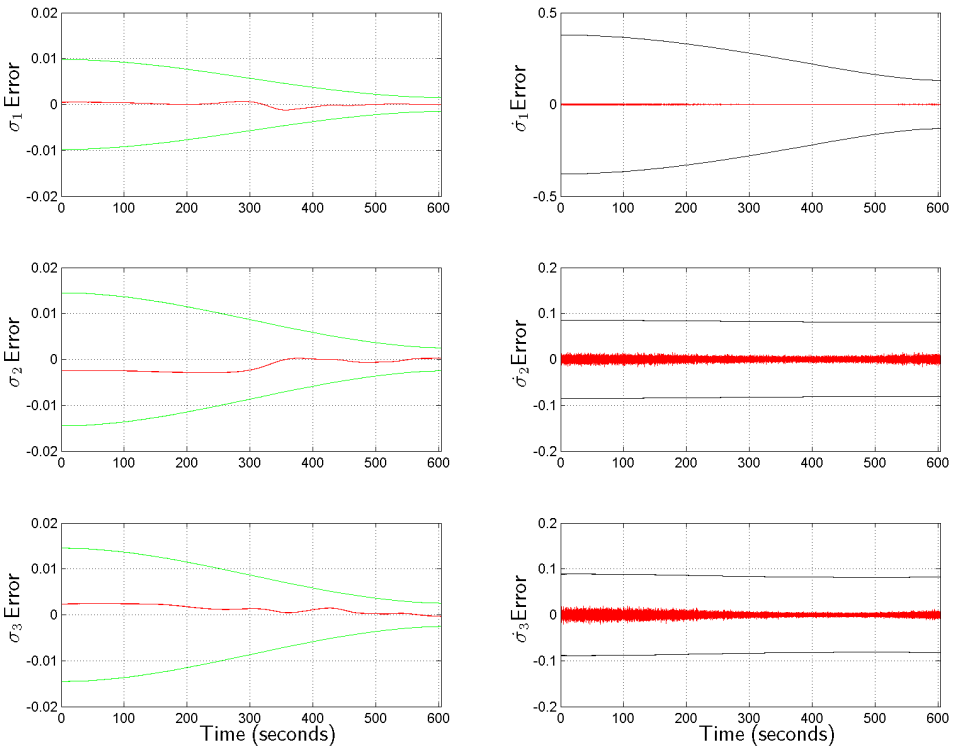

Fig. 133. Orientation & Attitude Rate Estimate Errors, Redone Max-Min-Min Attitude Case

relative attitude error in both pitch and roll simultaneously. This produced chaser relative attitude initial conditions of 7.5° of deviation from the target vehicle angles in each axis. (The choice of 7.5° for the angles in this test case is discussed in Subsection 8.) The remaining simulation parameters such as chaser mass, chaser moments of inertia, and initial starting position were kept nominal for the duration of the test case. Note that the VisNav sensor and Kalman filter introduced sensor noise into the test case as well, as for the other cases.

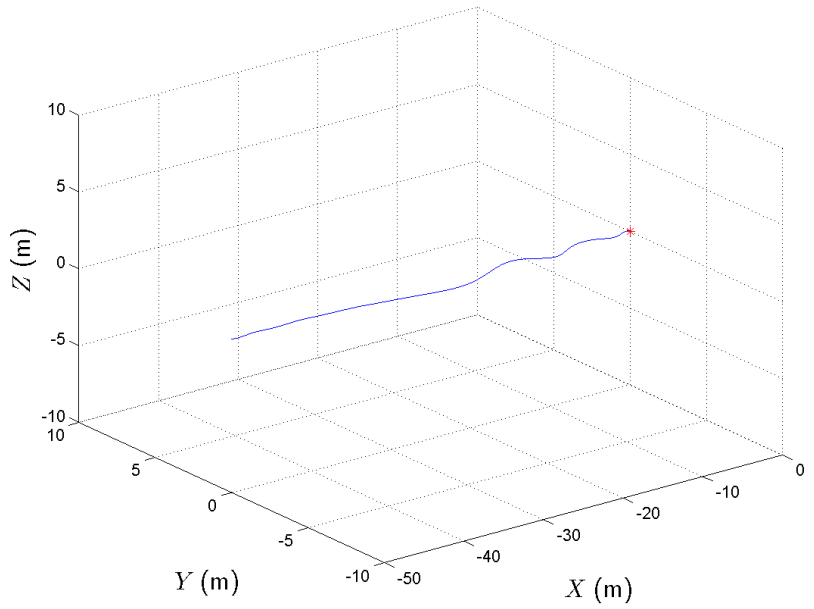

Fig. 134. Chaser Relative Trajectory, Min-Max-Max Attitude Case

Figure 134 shows the target frame relative trajectory for the current "min-max-max attitude" case. This trajectory appears to have the most deviation from nominal of any of the test cases yet. Inspection of the time history plots for the chaser relative states (shown in Figures 119 and 120) confirm that, like the previous test case, there are oscillations in y and z position that the controller does not adequately damp out in time for docking, and yaw and pitch also do not converge as well as in other cases.

Thus this case will also qualify as a successful dock "by the numbers" as the last one did, but again, qualitatively the controller's performance leaves something to be desired.

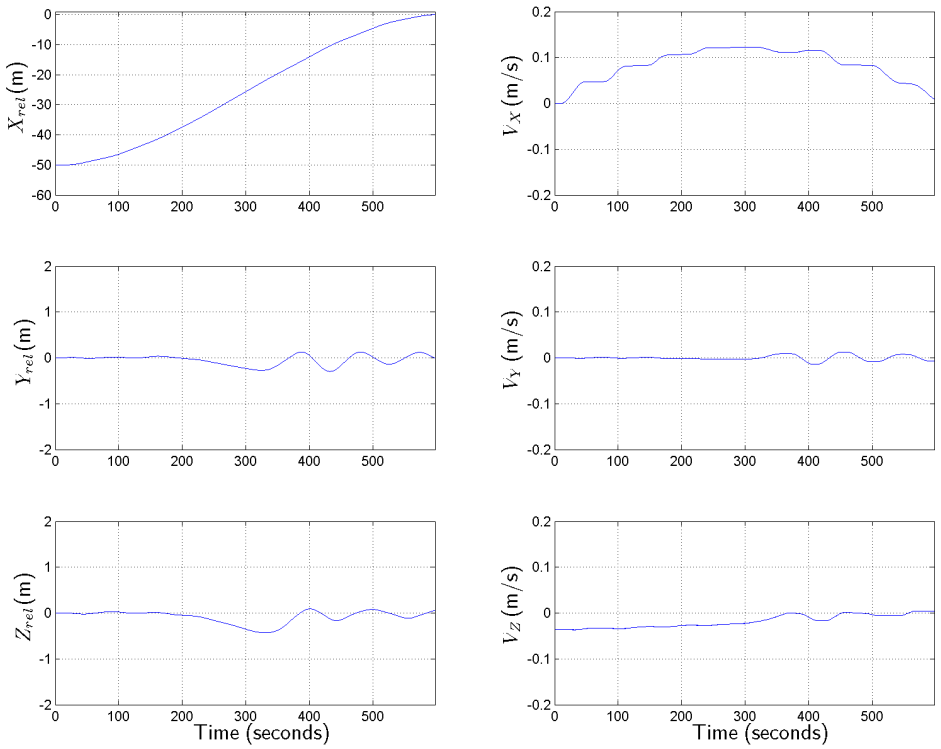

Fig. 135. Chaser Relative Position & Velocity, Min-Max-Max Attitude Case

The total position error at docking was 5.82 cm for this case, while the final attitude errors in yaw, pitch, and roll were -0.98, 0.73, and 0.37 degrees respectively. Docking was achieved for this case at 597.47 seconds. All of these final values for position, attitude, and time eclipsed the corresponding criteria for a successful dock, and the final attitude errors also (barely, for yaw) exceeded their "preferred" criteria.

The total relative velocity magnitude time history for the min-max-max attitude case is displayed in Figure 137. The total velocity appears to behave favorably for

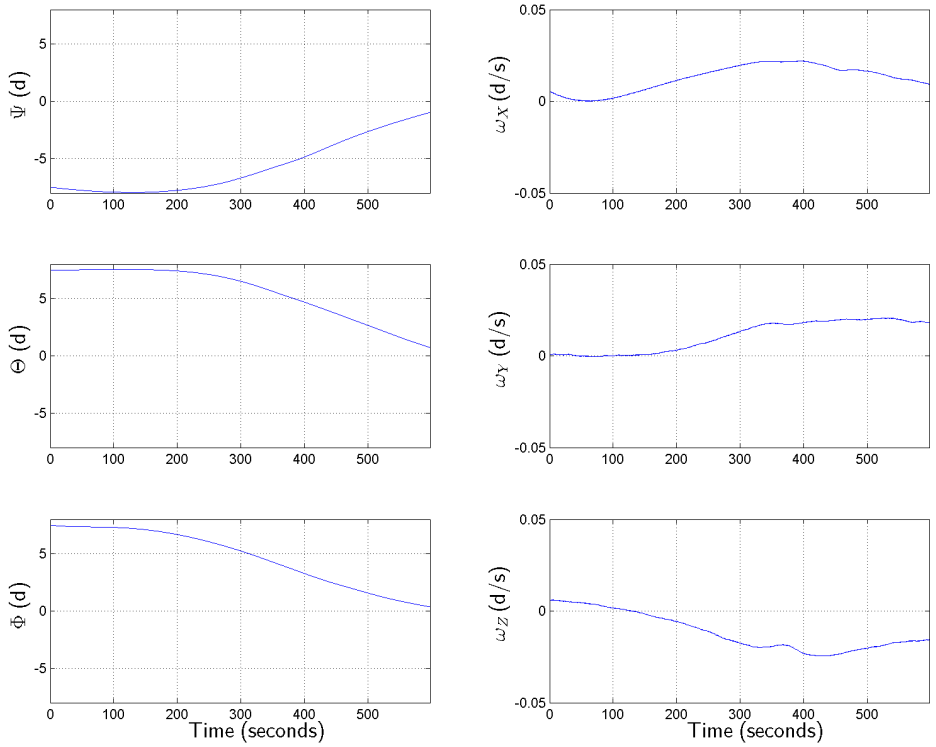

Fig. 136. Chaser Relative Orientation & Attitude Rate, Min-Max-Max Attitude Case

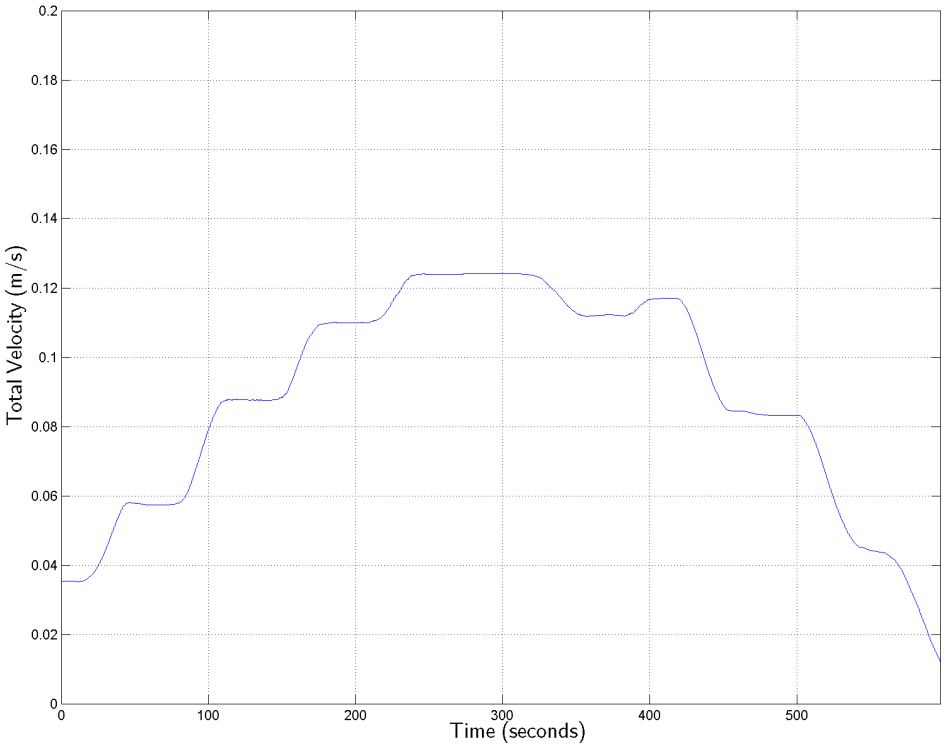

Fig. 137. Relative Velocity Magnitude Profile, Min-Max-Max Attitude Case

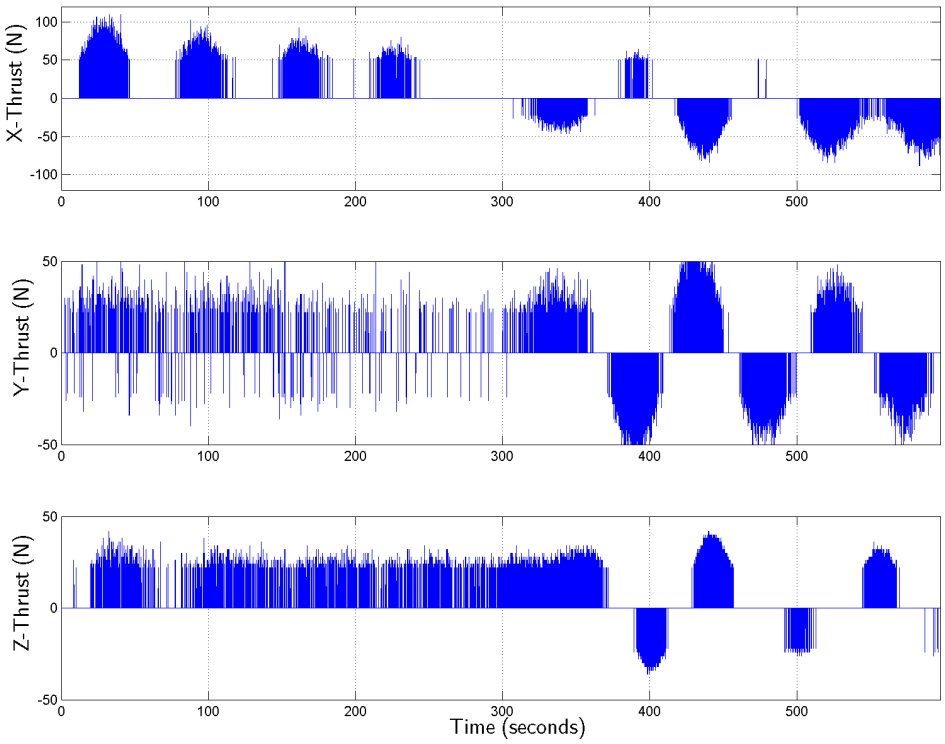

Fig. 138. Chaser Thrust Profile, Min-Max-Max Attitude Case

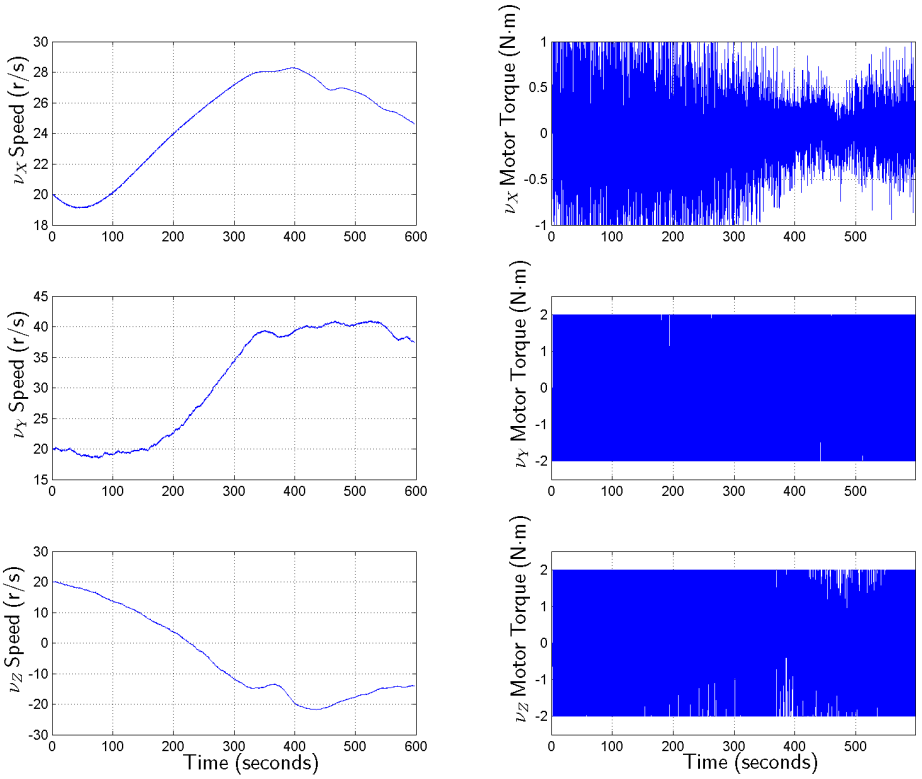

Fig. 139. Wheel Speeds & Motor Torques, Min-Max-Max Attitude Case

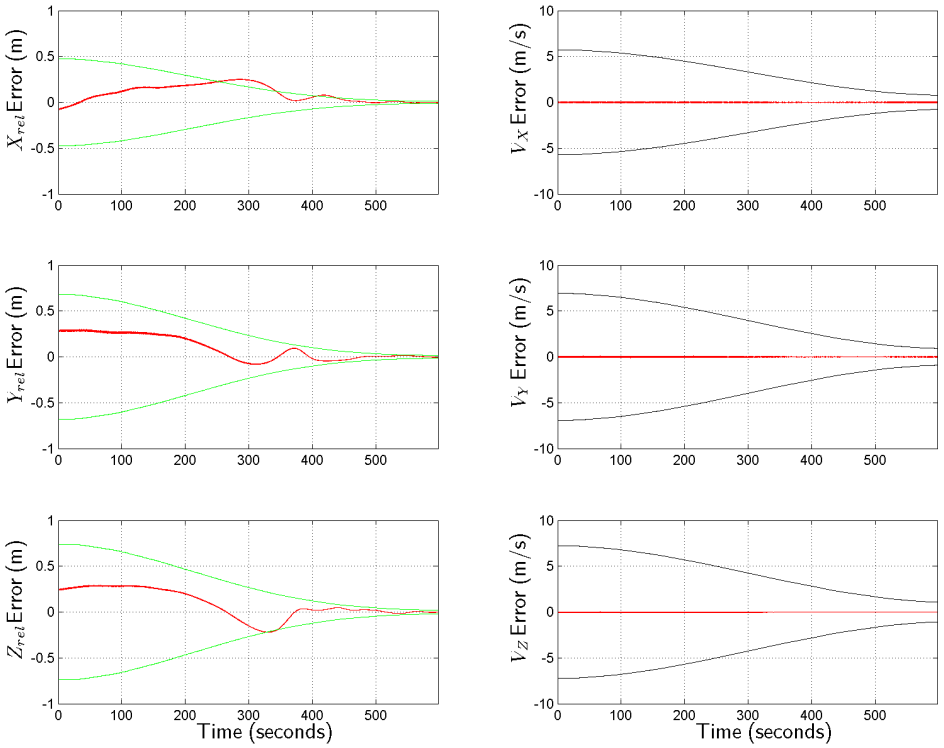

Fig. 140. Position & Velocity Estimate Errors, Min-Max-Max Attitude Case

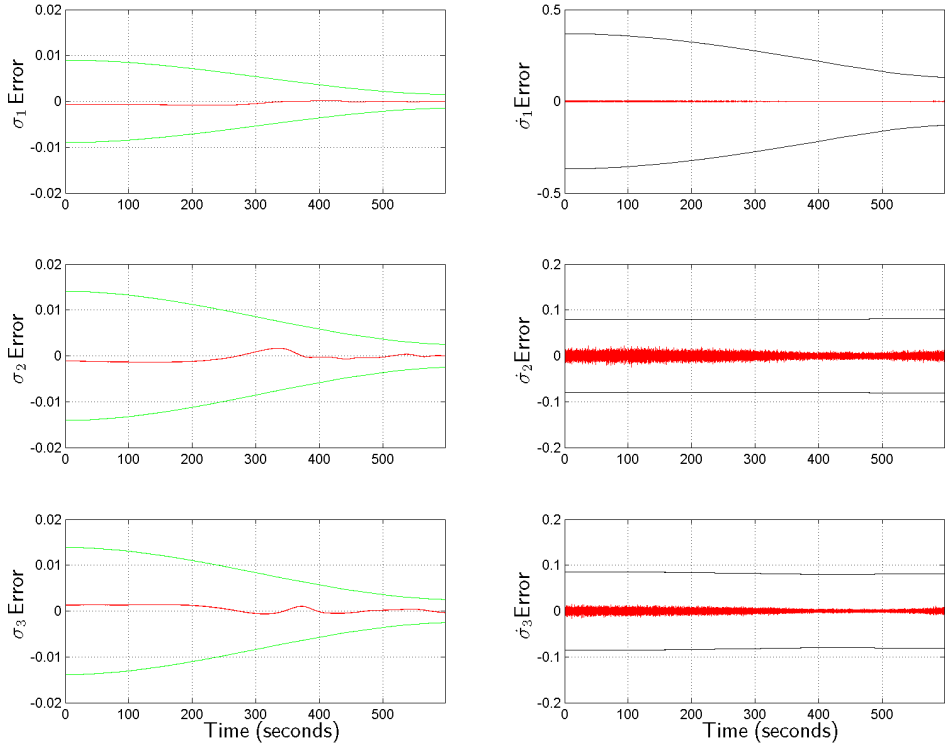

Fig. 141. Orientation & Attitude Rate Estimate Errors, Min-Max-Max Attitude Case

the duration of the run, though it exhibits more characteristic "wobble" than the last case due to the simultaneous oscillations in y and z. The final total velocity at dock was 1.24 centimeters per second, which bettered both the minimum and preferred docking criteria of 10 cm/s and 5 cm/s respectively. Therefore, based on the attitude and total velocity performance, this test run qualifies as "exceptional" strictly by the numbers according to this experiment's established criteria. The other plots for this "min-max-max attitude" case, shown in Figures 138 through 141, are consistent with the ones already examined here, and therefore will not be specifically discussed.

Since the controller met all docking success criteria but did not qualitatively perform as well as would be desired, as for the last case further investigations were performed to determine the root cause and attempt to improve upon the controll's baseline behavior in this case.

Studying the results just presented more closely, it became apparent that the inability of the controller to completely damp out the oscillations in y and z position and converge yaw and pitch stemmed from the same causes as the last case. So, a re-run of this test case was also conducted with the 5% deadband removed from the thruster system (to increase the available position fine control), and the saturation limits removed from the reaction wheel drive motors (to increase attitude control authority). This proved that the observations were correct: the position transients in y and z were adequately damped out before docking, and the attitude convergence was improved for yaw and pitch. However, the attitude still did not completely converge, as before. So, finally, the attitude gains in the controller were increased as well (to the same levels as for the last case, incidentally), and the controller system's performance then became fully satisfactory. Figures 142 through 149 summarize the improved results for this test run, which shows that these issues were in fact the source of the less than desirable performance by the baseline controller #2 in this case.

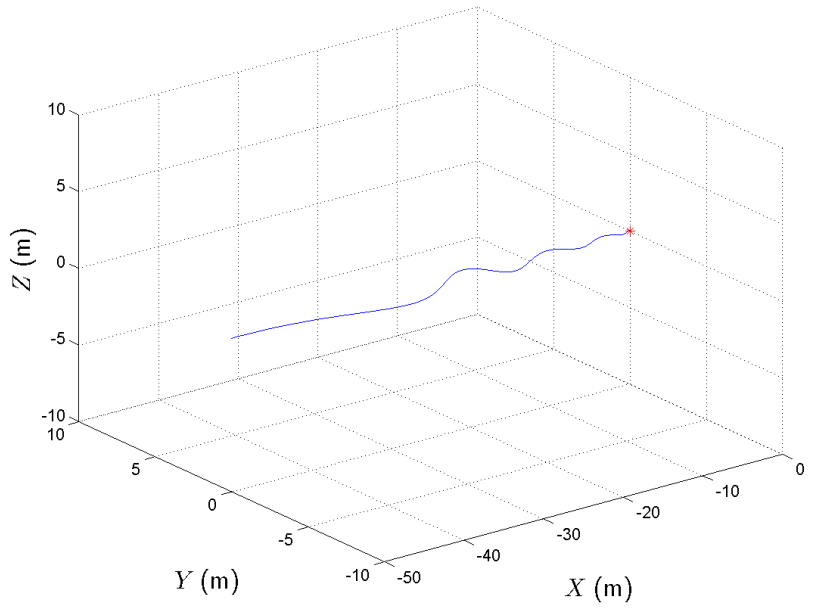

Fig. 142. Chaser Relative Trajectory, Redone Min-Max-Max Attitude Case

The total position error at docking for this modified case was 5.66 cm, a slight improvement over the baseline performance. Final attitude errors were -0.68, 0.51, and 0.36 degrees respectively in yaw, pitch, and roll, which also were improved as compared to the baseline. The docking maneuver completed for this case at 598.41 seconds (an improvement over the baseline as well), and the final total velocity at docking was 1.56 centimeters per second. These final values all exceeded their respective criteria for a successful dock, with the final attitude errors and total final velocity all exceeding their assigned "preferred" criteria.

Discussion of these results in the context of the entire controller #2 portion of the experiment will be performed in Sub-section 14.

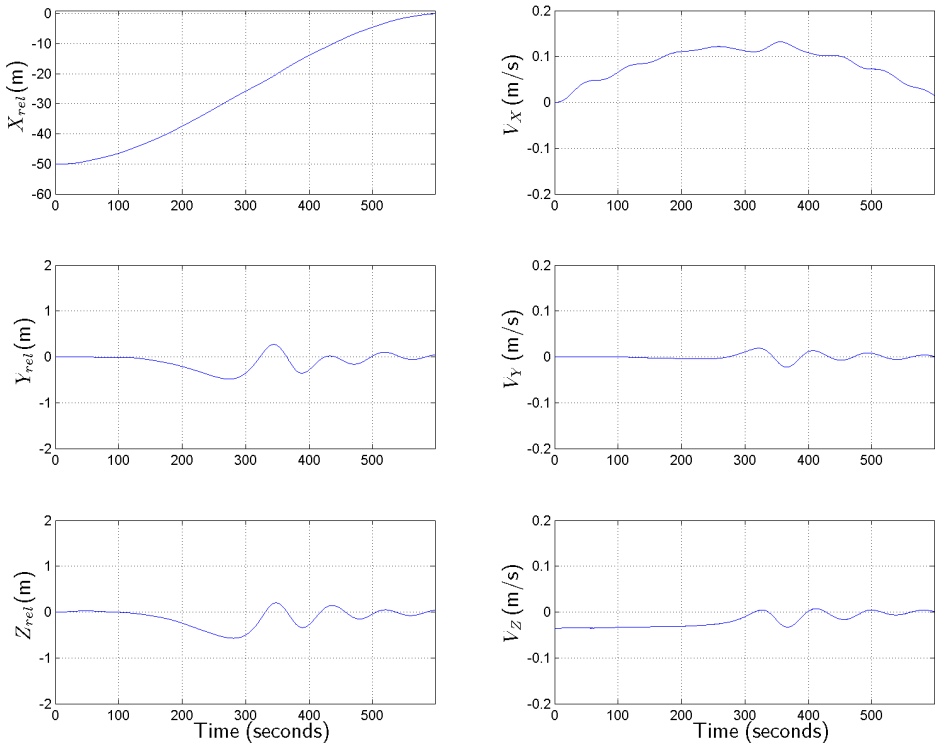

Fig. 143. Chaser Relative Position & Velocity, Redone Min-Max-Max Attitude Case

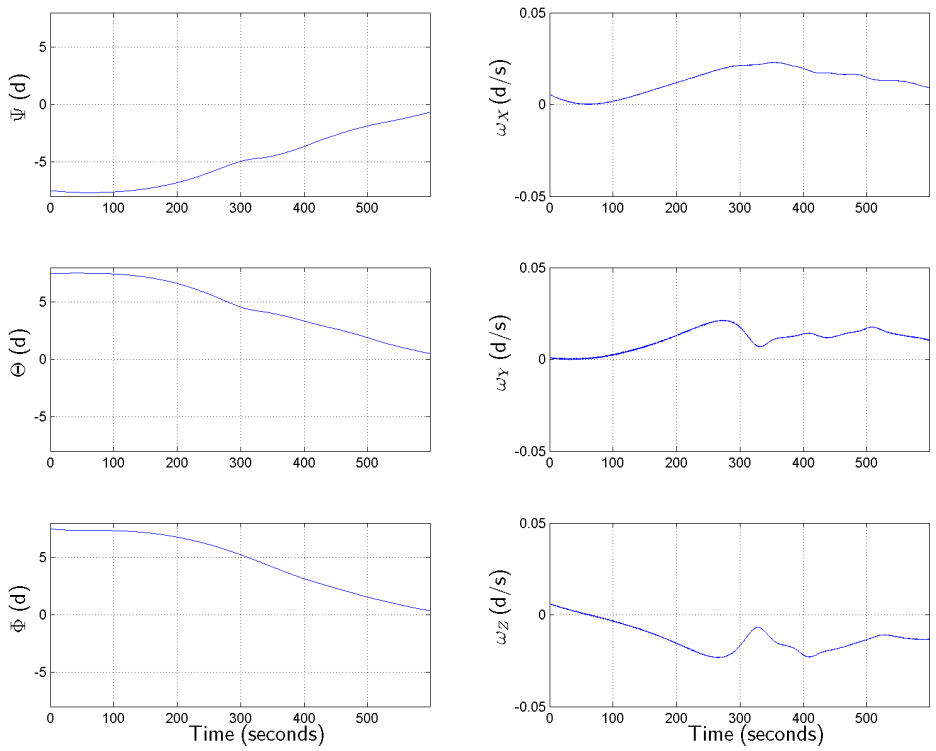

Fig. 144. Chaser Relative Orientation & Attitude Rate, Redone Min-Max-Max Attitude Case

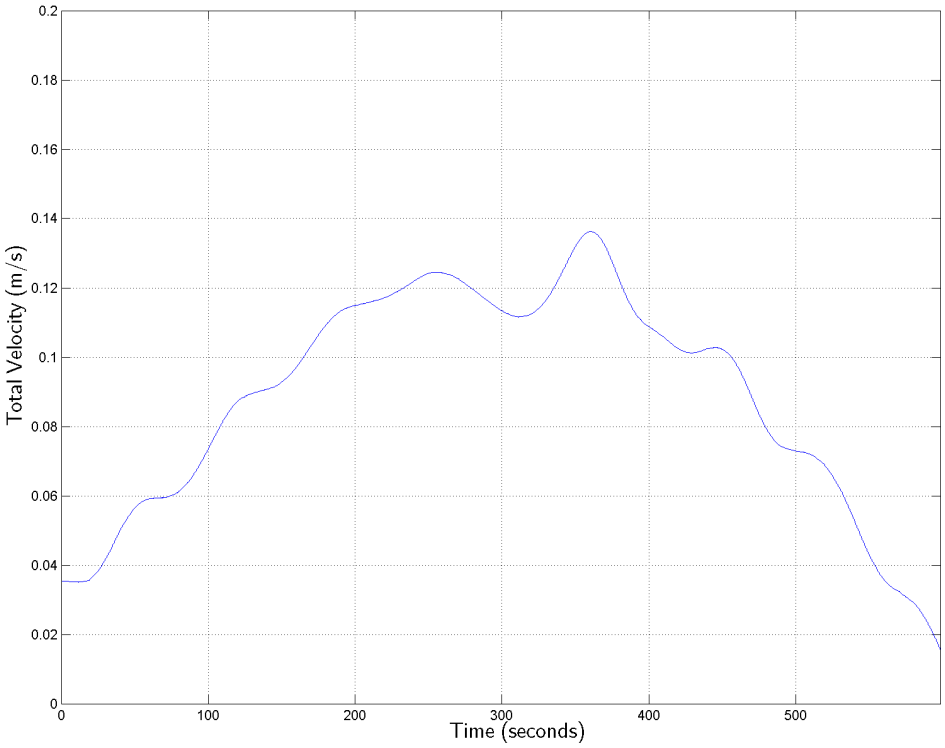

Fig. 145. Relative Velocity Magnitude Profile, Redone Min-Max-Max Attitude Case

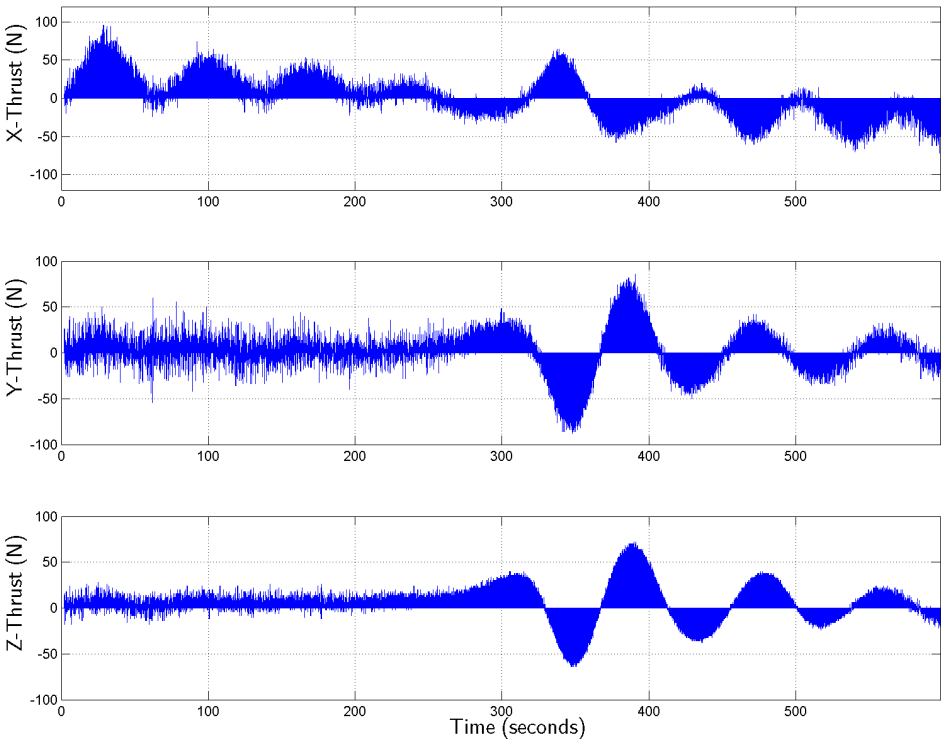

Fig. 146. Chaser Thrust Profile, Redone Min-Max-Max Attitude Case

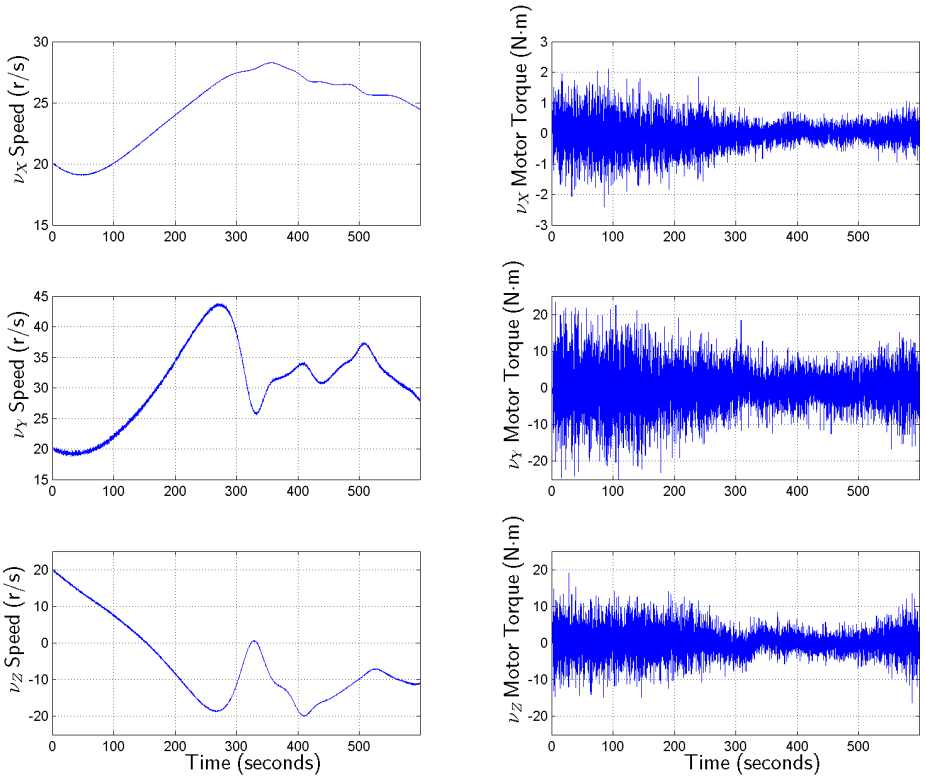

Fig. 147. Wheel Speeds & Motor Torques, Redone Min-Max-Max Attitude Case

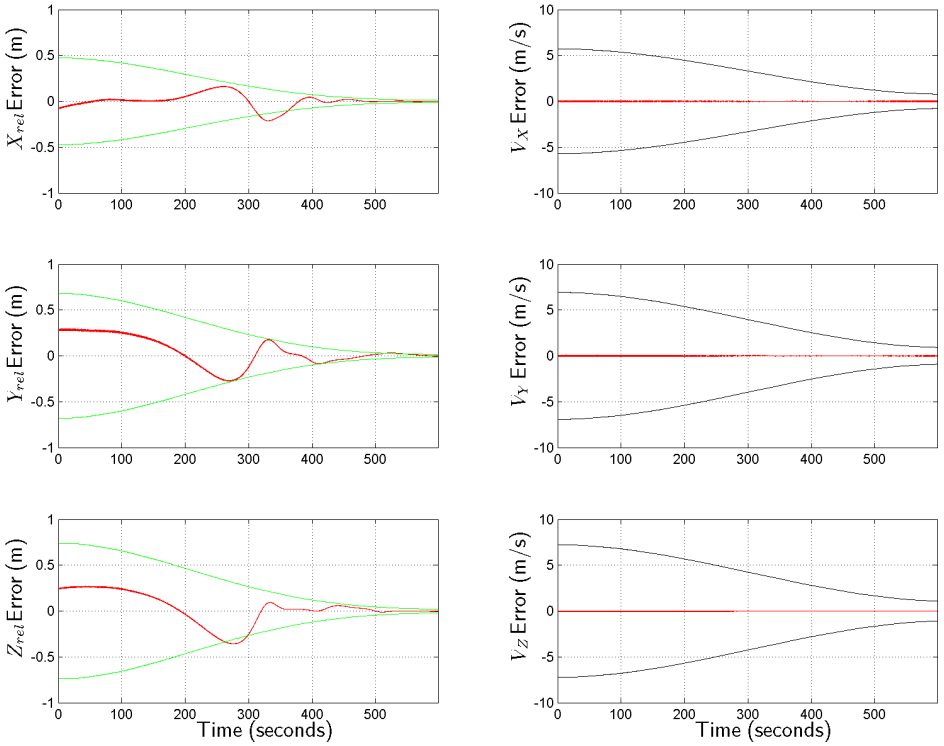

Fig. 148. Position & Velocity Estimate Errors, Redone Min-Max-Max Attitude Case

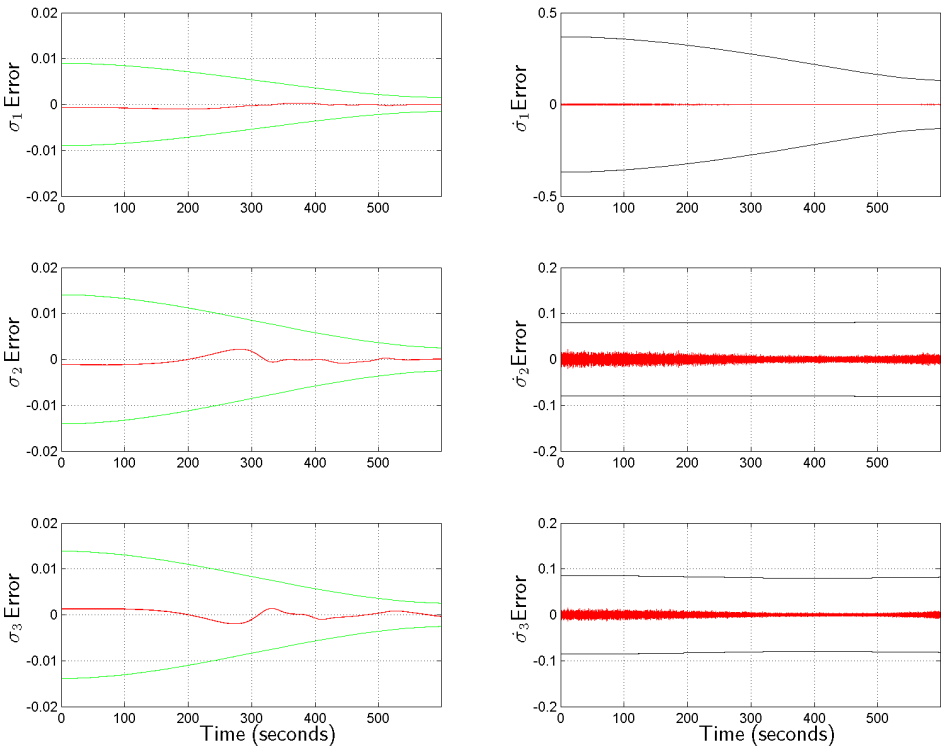

Fig. 149. Orientation & Attitude Rate Estimate Errors, Redone Min-Max-Max Attitude Case

12. Test Case 11: Max Positive Mass Uncertainty

Test Case 11 for controller #2 evaluated its performance robustness when given a maximum *positive* chaser mass error. For the controller #1 test plan the 3-σ value for mass variation was 15%; so to be consistent, this test case added 15%, or 2,850 kg, to the nominal value of the chaser's mass (19,000 kg). All other conditions for the test run were the same as the nominal case—including chaser moments of inertia, initial relative attitude, and initial starting position; there was also sensor noise from VisNav and the Kalman filter just as in the nominal.

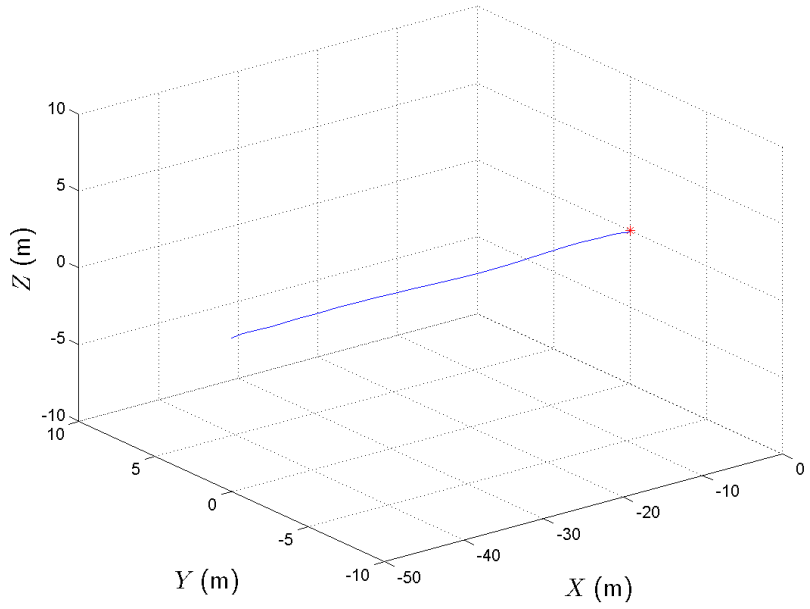

Fig. 150. Chaser Relative Trajectory, Max Mass Case

The target frame relative trajectory for the "max mass" test case is found in Figure 150. Note that its behavior looks very similar to the nominal; the controller seems to have accomodated the maximum 3-σ mass increase without issue. This implies a greater robustness to mass variation than what was exhibited by controller

#1, which could not compensate adequately for such large mass variations.

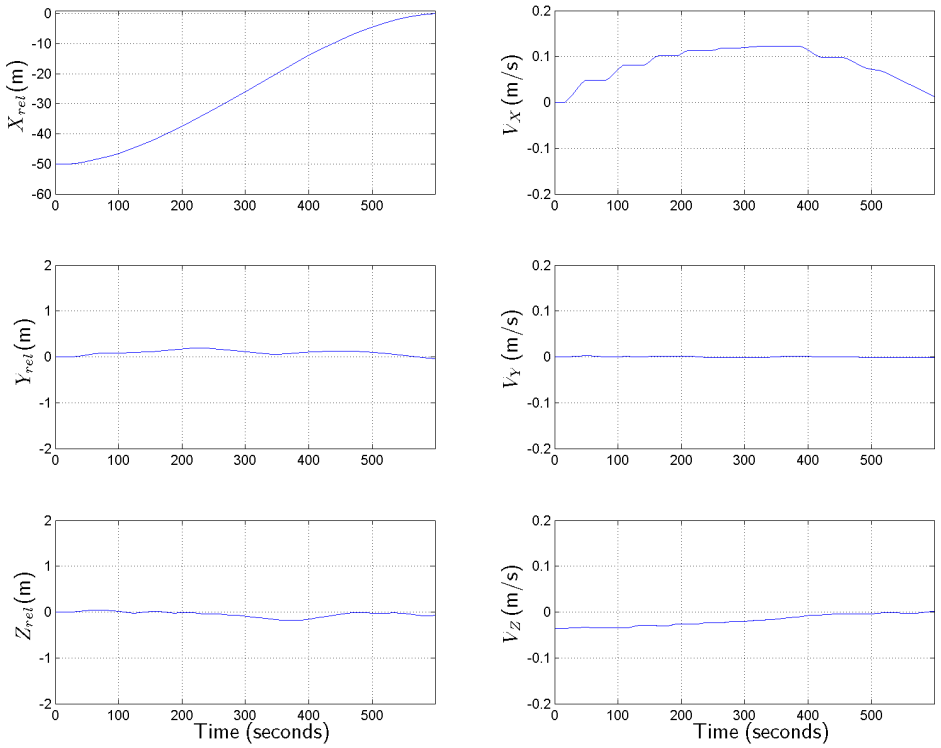

Fig. 151. Chaser Relative Position & Velocity, Max Mass Case

The Test Case 11 relative states are shown in Figures 151 and 152. These plots too appear to resemble the nominal results very closely, in spite of the large difference in chaser mass. This is a pleasing result, since it implies that the controller would work just as well for a heavily loaded vehicle as for the nominal. The final total position error for this test case is 8.29 cm, while the final attitude errors are 0.02, -0.28, and 0.19 degrees respectively in yaw, pitch, and roll—all of which easily eclipsed the required values for a successful dock, as well as the "preferred" docking criteria. Finally, the docking was completed in 599.18 seconds of elapsed simulation time, also easily exceeding the time requirement for success.

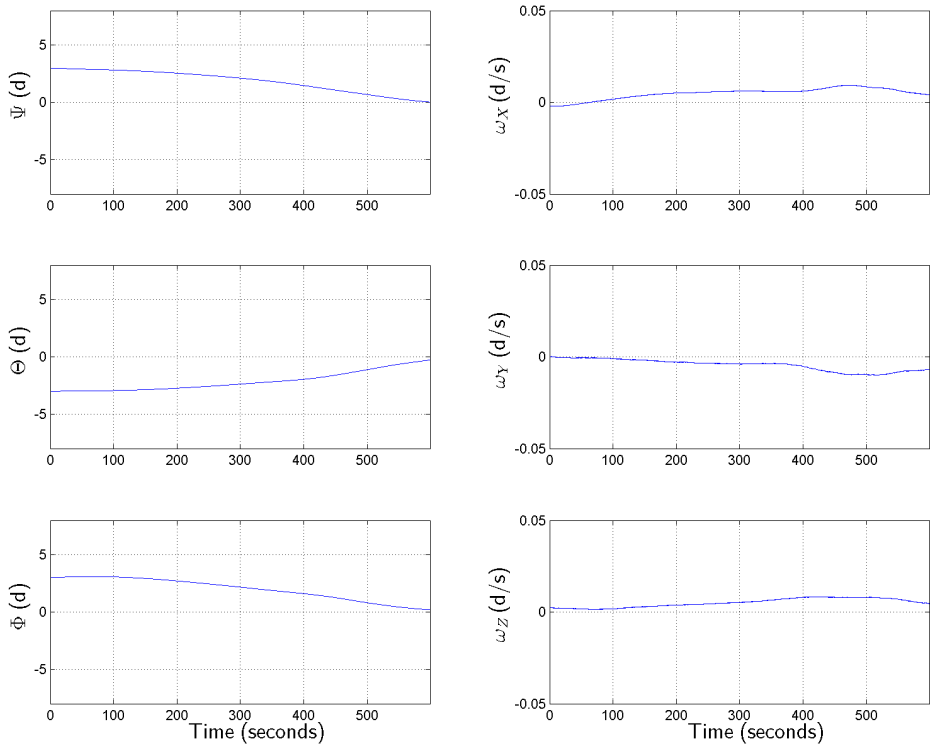

Fig. 152. Chaser Relative Orientation & Attitude Rate, Max Mass Case

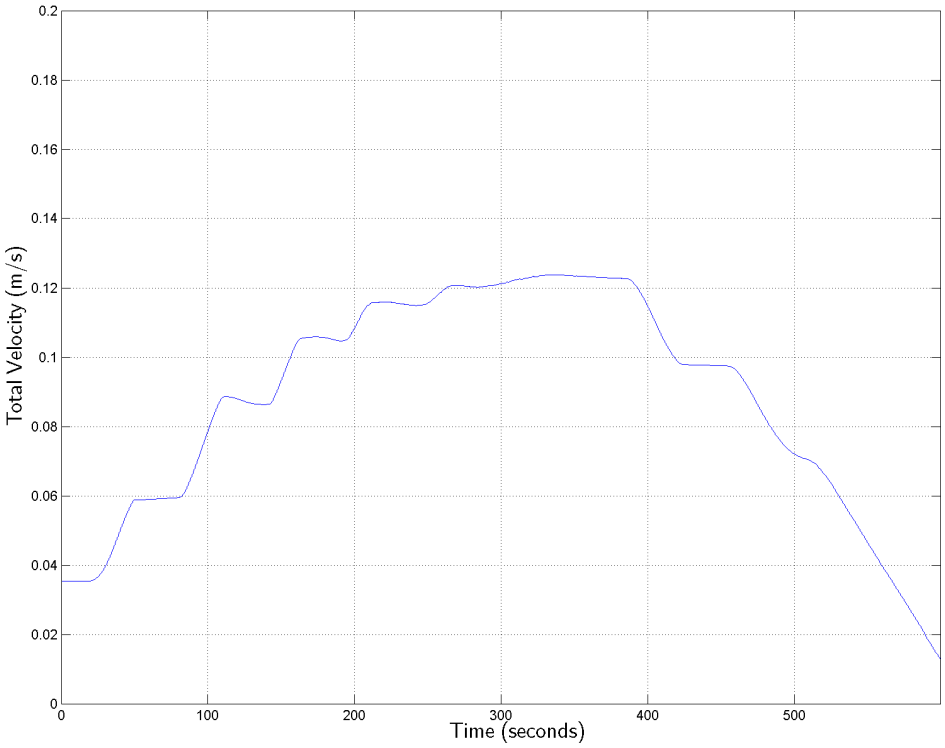

Fig. 153. Relative Velocity Magnitude Profile, Max Mass Case

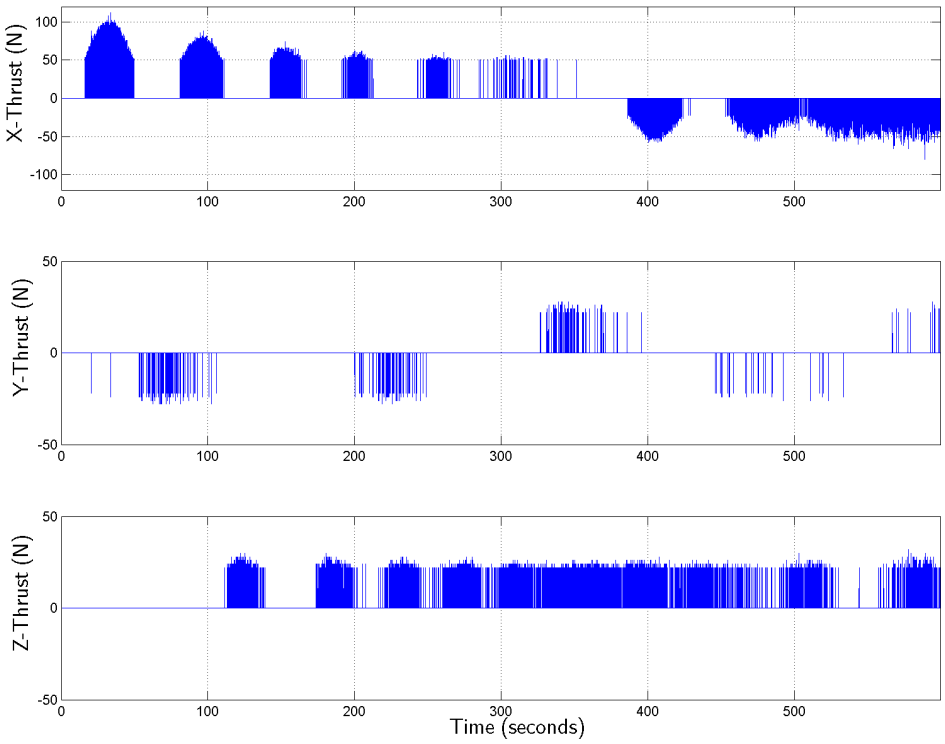

Fig. 154. Chaser Thrust Profile, Max Mass Case

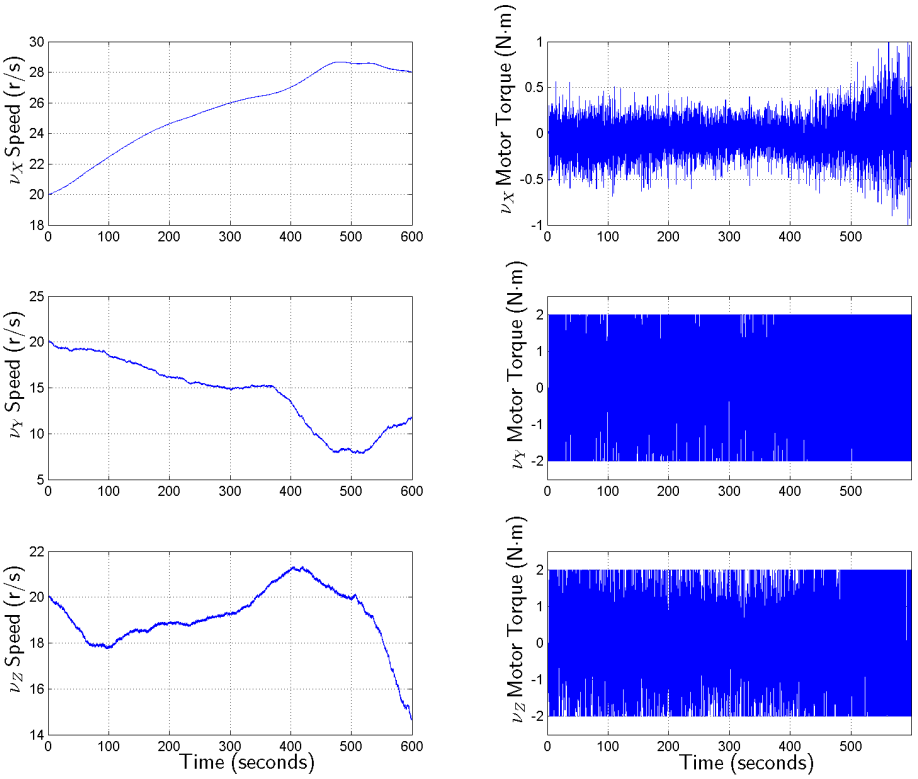

Fig. 155. Wheel Speeds & Motor Torques, Max Mass Case

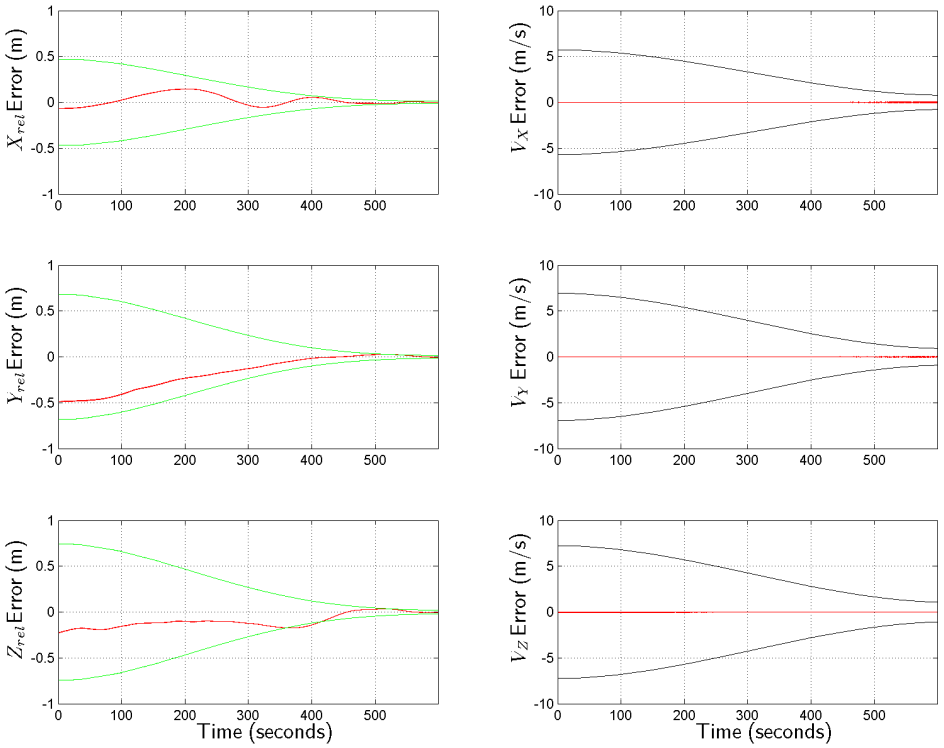

Fig. 156. Position & Velocity Estimate Errors, Max Mass Case

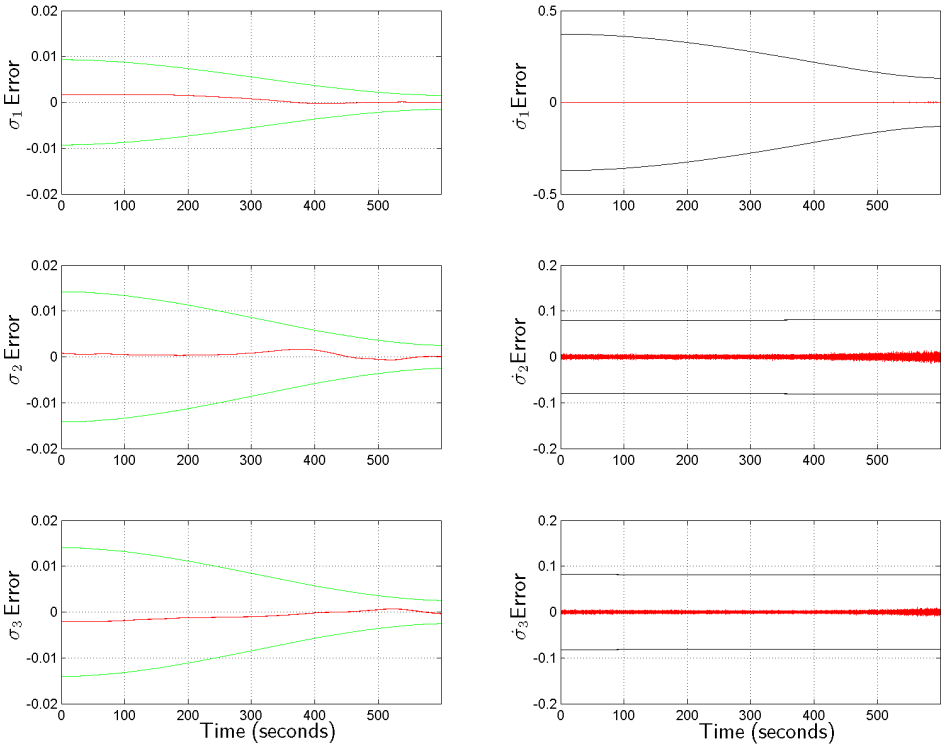

Fig. 157. Orientation & Attitude Rate Estimate Errors, Max Mass Case

Figure 153 shows the total relative velocity magnitude time history for this test case. While it perhaps varies slightly more than the nominal one, it still exhibits very desirable behavior particularly near the end. The velocity magnitude at docking is 1.30 centimeters per second, which is well below both the desired velocity limit (10 cm/s) and the preferred limit (5 cm/s). Based on this and the final relative attitude at dock, the "max mass" test case is considered an "exceptional" run as well.

The rest of the results for the "max mass" test case for controller #2 are presented in Figures 154 through 157. The graphs are very similar to the nominal plots and thus in essence speak for themselves. They show, in short, that the controller easily handled the positive 3-σ mass error case, with results very similar to the nominal; the VisNav sensor and the Kalman filter also performed satisfactorily throughout this test run.

These "max mass" case results will be briefly discussed in the context of the full controller #2 experiment in Sub-section 14.

13. Test Case 12: Max Negative Mass Uncertainty

The final test case for controller #2 evaluated its performance robustness against a maximum *negative* chaser mass error. The controller #1 test plan used a mass variation 3-σ value of 15%; so, this test case subtracted 15%, or $2,850$ kg, from the chaser vehicle's nominal mass ($19,000$ kg). All remaining conditions were held at nominal values throughout the run—including chaser moments of inertia, initial relative attitude, and initial chaser relative starting position. There was also sensor noise from VisNav and the Kalman filter, as was present in all test cases.

The relative trajectory, expressed in the target frame, for the "min mass" case is shown in Figure 158. Note that its behavior looks very similar to the nominal just as the "max mass" one did; the controller seems to handle the maximum negative 3-σ

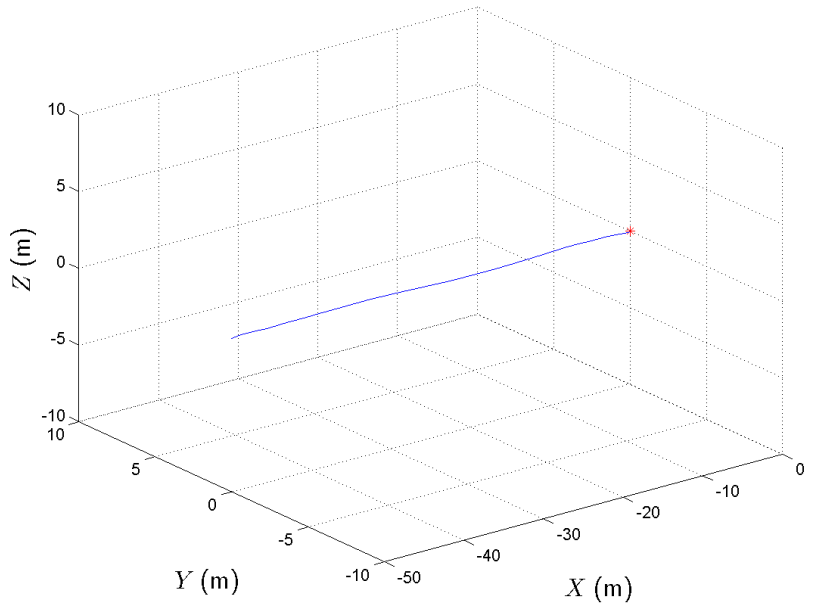

Fig. 158. Chaser Relative Trajectory, Min Mass Case

mass error as well as it did maximum positive 3-σ mass error case. This confirms that controller #2 has greater mass robustness than controller #1, since it can accomodate larger mass variations in both (positive and negative) directions.

The Test Case 12 relative state versus time plots are in Figures 159 and 160. These plots have very similar characteristic behavior to the nominal case, in spite of the reduced mass. This too is a favorable result, since it implies that the controller would work just as well for a much lighter vehicle as it did for the nominal. The final total position error for the "min mass" case is 5.61 cm, with final attitude errors of 0.02, -0.24, and 0.18 degrees at docking, in yaw, pitch, and roll respectively. These values each clearly and easily exceeded the criteria for a successful dock, as well as the preferred final value criteria. Finally, the docking was completed at time $t = 601.44$ seconds, which also easily surpassed its success criteria.

Figure 161 shows a plot of the Test Case 12 total relative velocity magnitude

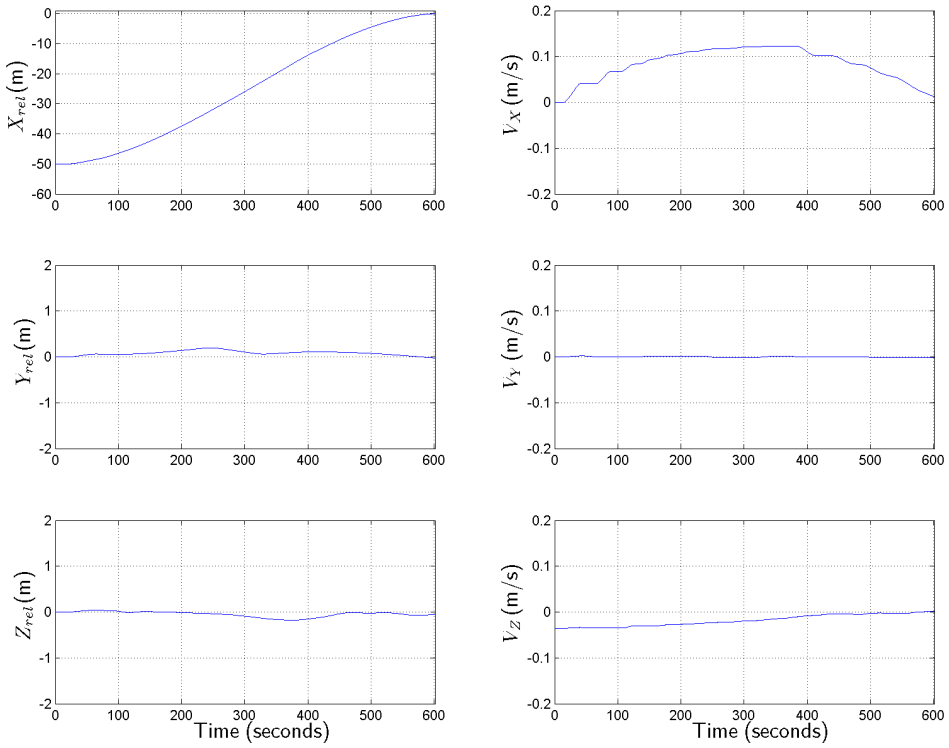

Fig. 159. Chaser Relative Position & Velocity, Min Mass Case

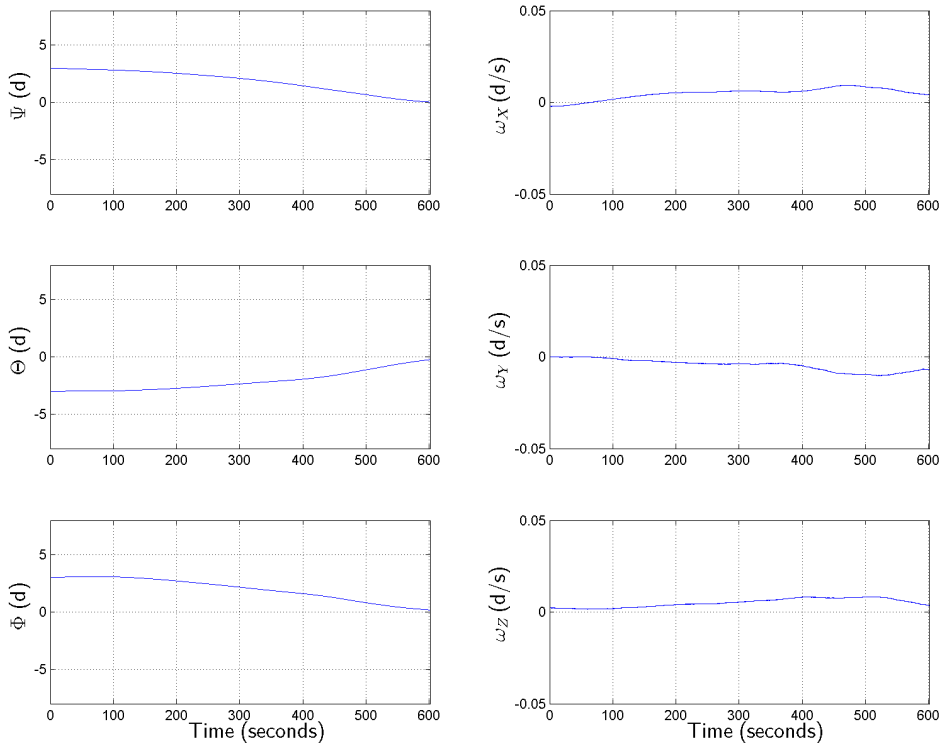

Fig. 160. Chaser Relative Orientation & Attitude Rate, Min Mass Case

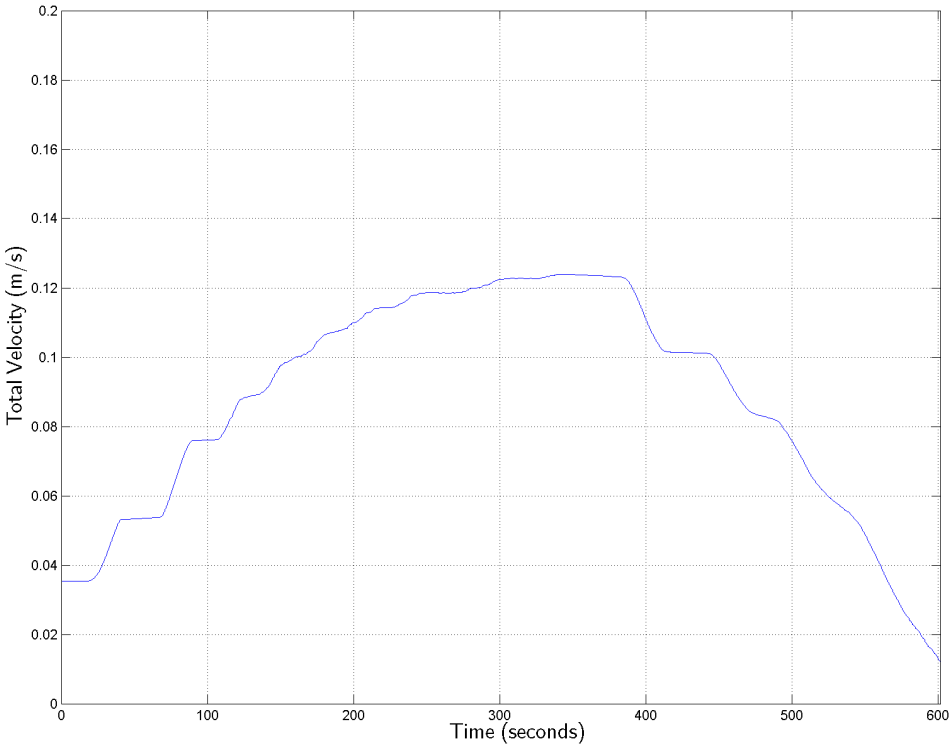

Fig. 161. Relative Velocity Magnitude Profile, Min Mass Case

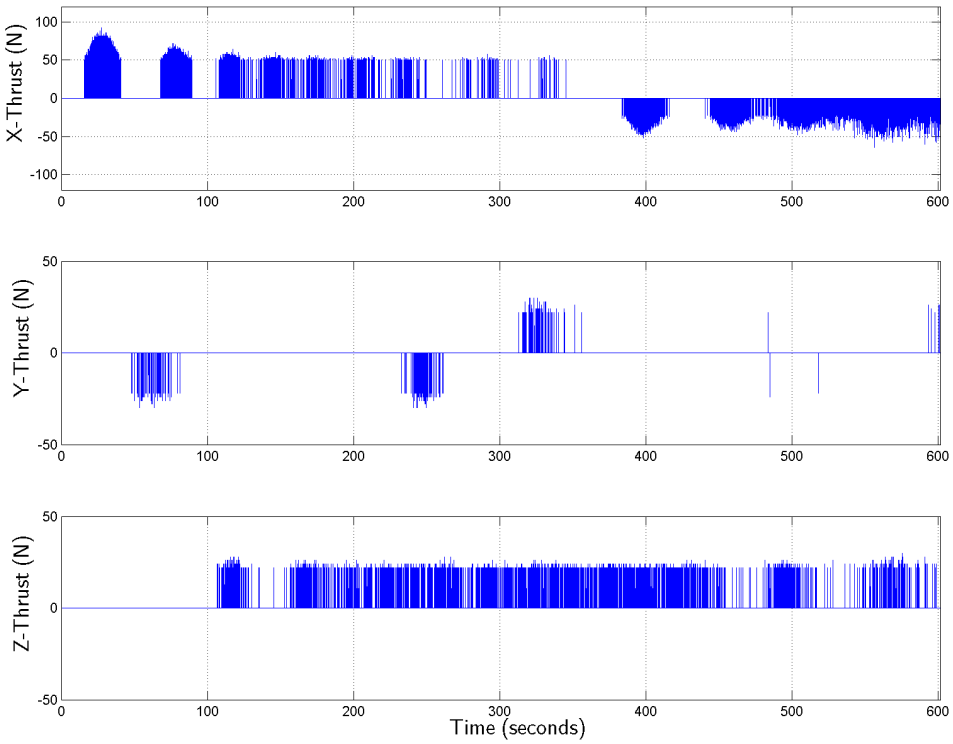

Fig. 162. Chaser Thrust Profile, Min Mass Case

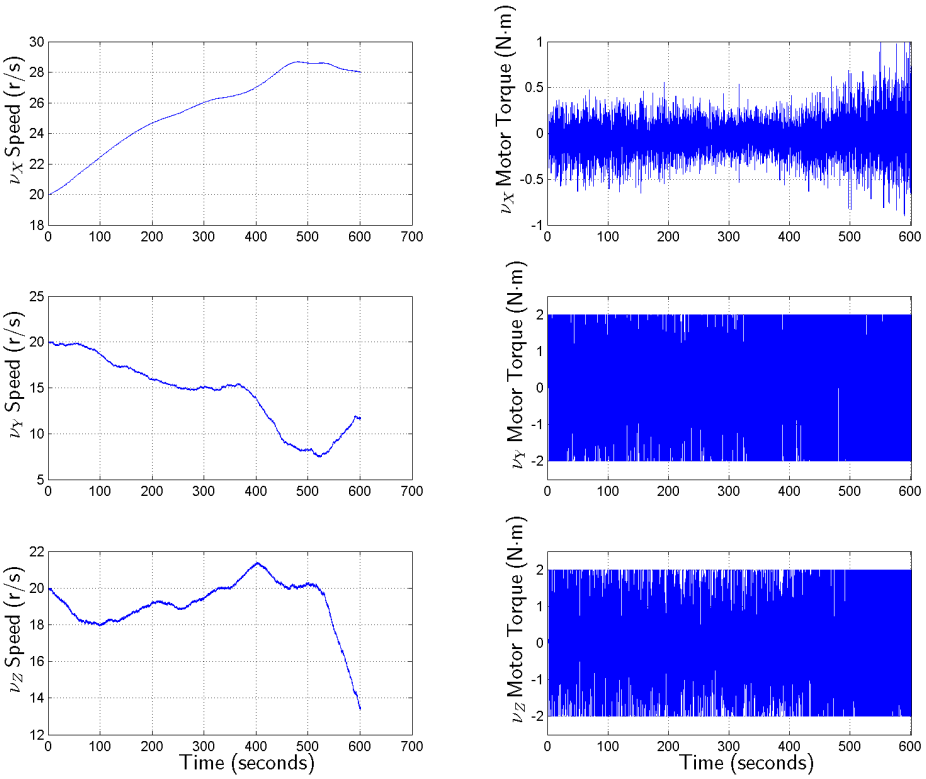

Fig. 163. Wheel Speeds & Motor Torques, Min Mass Case

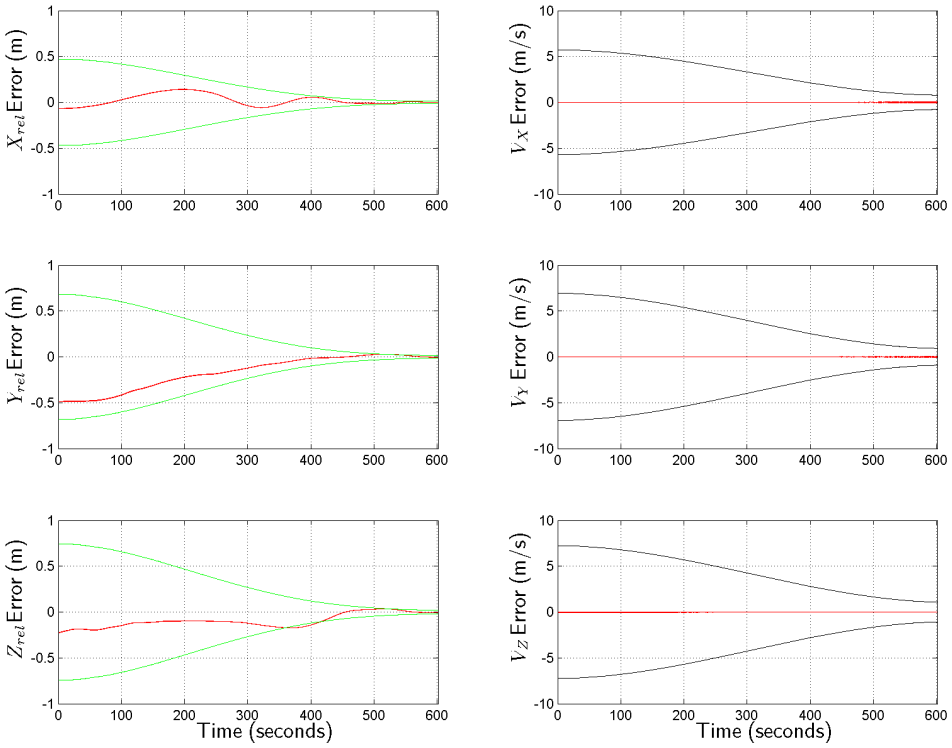

Fig. 164. Position & Velocity Estimate Errors, Min Mass Case

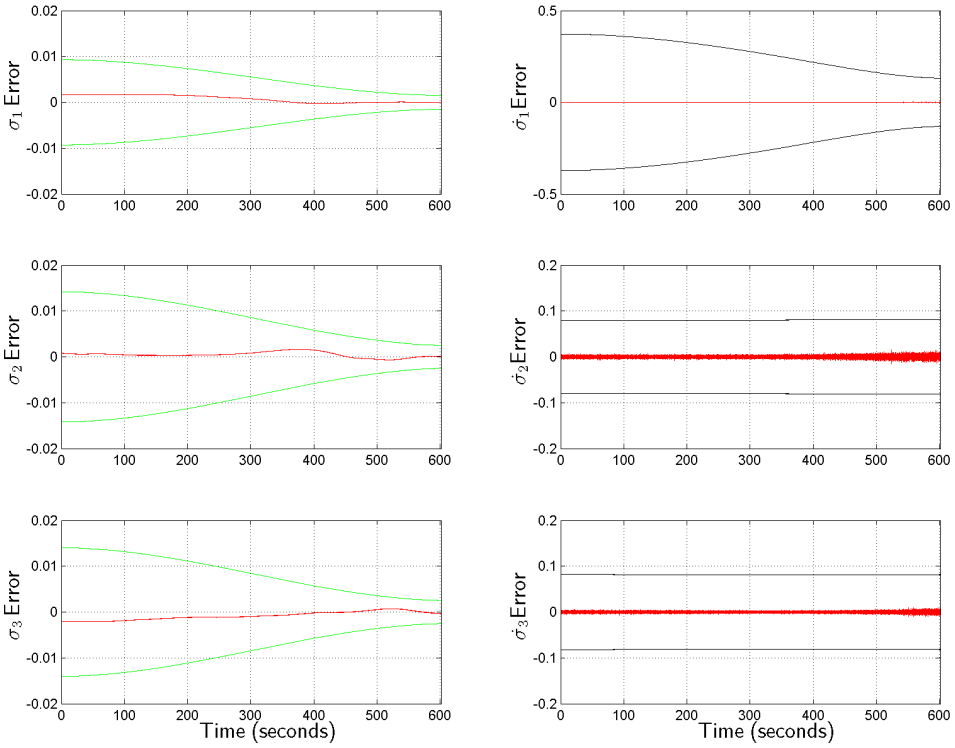

Fig. 165. Orientation & Attitude Rate Estimate Errors, Min Mass Case

versus time. The graph shows very desirable general behavior of the total velocity, with a magnitude at docking of about 1.22 centimeters per second. This value too is well below both the desired and preferred velocity limits of 10 cm/s and 5 cm/s respectively. Thus, this test case is an "exceptional" run as well according to this experiment's criteria for that designation.

The remaining results for the "min mass" test case for controller #2 are presented in the following figures (162 through 165). These graphs are so similar to those for the nominal case that no special discussion of them is warranted. The controller easily absorbed the negative 3-σ mass error condition and showed exceptional results in spite of it. Since the VisNav sensor and Kalman filter also performed well, the test case was generally a success on all points.

These test case results will be further discussed in the next sub-section, along with the rest of the controller #2 test runs.

14. Summary of Results

In contrast to the first controller, controller #2 performed very well in nearly every test case to which it was subjected, achieving "exceptional" results for all tests performed except the "max attitude" case. However, it should be noted that its performance was a bit lacking in both the max-min-min and min-max-max attitude cases, in spite of meeting the standard for "exceptional" results for both of those two cases. Further investigations revealed and confirmed the reasons behind this behavior, as was discussed in Sections 10 and 11. The overall controller #2 results are all the more remarkable considering that many of these runs were statistically unrealistic worst on worst on worst initial case conditions. Thus, the second controller is clearly superior to the first, and appears to be suitable for use in its current form as a docking controller for all reasonable scenarios likely to be encountered in a real world

application.

In addition to the dramatically better results delivered by the second controller, the VisNav and Kalman filter system also showed significantly improved performance over the first set of tests. This was mostly due to the discovery of a small but significant error in the implementation of the attitude portion of the filter, though the fact that the filter was retuned certainly played a part as well. Overall, the controller #2 tests serve to demonstrate the feasibility of using VisNav in concert with a Kalman filter for automated docking. These results give much greater optimism for its capabilities in "real world" docking scenarios than the first results did, and thus the second part of the experiment is considered a success from this standpoint as well.

CHAPTER IX

CONCLUSIONS

This thesis presented a system for performing automated docking of unmanned spacecraft, the system development process, and a performance evaluation of the system via simulation. The automated docking system uses a vision-based relative navigation sensor package named VisNav that generates accurate relative 6-DOF estimates at a 100 Hz update rate. The VisNav sensor was integrated with a Kalman post-filter that generated real-time relative velocity and relative angular velocity estimates for use by the docking controllers.

To evaluate controller performance in a realistic scenario, a computer simulation was developed of a spacecraft docking in lunar orbit. The chosen scenario consisted of the chaser vehicle starting in the same circular orbit as the target vehicle, trailing it at a nominal distance of 50 meters. The simulation used realistic dynamical vehicle models based on the European Automated Transfer Vehicle, an automated resupply shuttle that is currently being used to service the International Space Station. The controllers were evaluated by a series of parametric case studies, which examined performance robustness with respect to mass modeling uncertainty, moment of inertia modeling uncertainty, relative starting position errors, and relative initial attitude errors respectively. The simulation results showed that when using controller #2, the automated docking system could achieve docking reliably in spite of realistic sensor noise, even if any of the other types of uncertainty or error were also present. Also, the VisNav sensor and Kalman filter package's performance was found to be satisfactory for enabling successful docking with controller #2 due to good error convergence characteristics over the operating range considered.

Conclusions that are drawn based on results presented in this thesis include the

following:

1. Docking controller #2 is capable of docking with both high accuracy and high precision, even in the presence of realistic simulated measurement noise. In these nominal (for this test) conditions, the error was less than 7 centimeters at docking, with the controller easily exceeding all of the other constraints on the final conditions (time, docking velocity, and orientation angles) as well.

2. Controller #2's performance robustness was demonstrated for chaser vehicle modeling uncertainty; robustness was shown both for the case of mass errors and for errors in the moments of inertia. The precision and accuracy exhibited in the presence of these errors was not significantly degraded from the nominal performance, even for mass errors of up to $\pm 15\%$ and moment of inertia errors of up to $\pm 15\%$ per axis simultaneously.

3. The performance robustness of controller #2 was demonstrated for most relative initial condition errors. It exhibited robustness both for relative starting position errors and for some relative initial attitude errors, and its performance was not significantly degraded from the nominal even for errors of up to $\pm 15\%$ in position. Its performance with ± 7.5 deg per axis errors simultaneously in attitude was also generally satisfactory, though due to cross-coupling and thruster/reaction-wheel imposed limitations it struggled with the max-min-min and min-max-max combinations of attitude error.

4. VisNav with an integrated Kalman post-filter provided highly accurate relative navigation estimates over the approach distance considered and thus contributed to the automated docking system's success. This performance serves to demonstrate the basic feasibility of using this rel-nav package in a spacecraft

docking application.

CHAPTER X

RECOMMENDATIONS

This chapter presents recommendations for improvement, advancement, and further study based on the outcomes of this research. The three major areas of focus for these recommendations are the VisNav sensor system and sensor model, the docking controller developed in this work, and the medium-fidelity docking simulation created to support this project.

The VisNav sensor system continues to develop and mature. As it does, it also continues to show itself to be a capable relative navigation sensor worthy of consideration in real-world applications. To further improve its utility for spacecraft proximity operations, it is recommended that efforts be made to increase the operational range of the sensor. This would help the sensor to be more competitive against laser-based solutions such as lidar.

The VisNav sensor simulation model has already proven to be a useful tool for studying possible relative navigation applications of VisNav. As the sensor continues to develop, efforts should continue to ensure that the simulation model keeps pace with its advances so that this important modeling capability remains relevant. In the meantime, more rigorous and standardized hardware and simulation benchmarking procedures should be developed to facilitate clearer comparisons between model and hardware performance. This will increase the benefits of having the sensor simulation as a design and analysis tool.

The final docking controller designed in this work, controller #2, was found to be robust to two different types of vehicle modeling uncertainty for the given scenario—vehicle mass, and vehicle moments of inertia. It was also found to be robust to two different types of initial condition errors in the given scenario—relative initial position

error, and relative initial attitude error. However, the controller was tested with each of these types of uncertainty or error individually, which is of limited application to real-world spacecraft docking scenarios. Therefore, further investigations should be performed that examine the controller's performance in the presence of multiple types of uncertainty simultaneously. Also, while the controller worked well in the single docking scenario used in this test, it should be subjected to a variety of different spacecraft docking scenarios to gain confidence in it as an "all-purpose" docking controller.

Finally, the docking simulation architecture was intentionally written in a general way, because it has many structural components that would be common to any spacecraft relative navigation or proximity operations scenario. This tool should be further developed by increasing its fidelity. Specifically, the ability to model environmental disturbance forces should be added. Also, further partitioning of the code into separate modules would allow easier substitution of new vehicle models, disturbance models, and planetary orbital data into the simulation. Ideally, the user should be able to make these changes without even having to see the structural sections of code. Together, these improvements would make the simulation a powerful and useful tool for orbital dynamics and spacecraft relative navigation studies.

REFERENCES

[1] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems*, Boca Raton, FL: Chapman & Hall/CRC, Applied Mathematics and Nonlinear Science Series, 2004, pp. 283–292.

[2] R. E. Bowers, "Estimation Algorithm for Autonomous Aerial Refueling Using a Vision Based Relative Navigation System," Master's thesis, Texas A&M University, College Station, TX, 2005.

[3] European Space Agency (October 2006), "Automated Transfer Vehicle," website. Available: www.esa.int/SPECIALS/ATV/index.html.

[4] M. Romano, "An on-the-ground simulator of autonomous docking and spacecraft servicing for research and education," in *Proc. SPIE Defense and Security Symposium - Spacecraft Platforms and Infrastructure Conference*, Orlando, FL, April 2004, SPIE article 5419-19.

[5] M. E. Polites, "Technology of automated rendezvous and capture in space," *Journal of Spacecraft and Rockets*, vol. 36, no. 2, pp. 280–291, March-April 1999.

[6] E. C. Ezell and L. N. Ezell, *The Partnership: A History of the Apollo-Soyuz Test Project*, Washington, D.C.: National Aeronautics and Space Administration, 1978, p. 94.

[7] M. E. Polites, "An assessment of the technology of automated rendezvous and capture in space," Marshall Space Flight Center, Huntsville, AL: NASA Technical Publication 1998-208528, July 1998.

[8] I. Kawano, M. Mokuno, T. Kasai, and T. Suzuki, "Result of autonomous rendezvous docking experiment of engineering test satellite-VII," *Journal of Spacecraft and Rockets*, vol. 38, no. 1, pp. 105–111, January-February 2001.

[9] Japan Aerospace Exploration Agency Press Release (February 2009), "HTV: An indispensable part of ISS operations," Available: www.jaxa.jp/article/special/transportation/torano01_e.html.

[10] NASA Press Release 05-051 (April 2005), "On-orbit anomaly ends DART mission early," Available: www.nasa.gov/lb/mission_pages/dart/media/05051.html.

[11] DARPA Tactical Technology Office (October 2006), "Orbital Express Space Operations Architecture Program," website. Available: www.darpa.mil/tto/programs/oe.htm.

[12] ESA Press Release 20-2008 (April 2009), "Europe's automated ship docks to the ISS," Available: www.esa.int/SPECIALS/ATV/SEM-ORO5QGEF_0.html.

[13] M. Cislaghi, M. Lellouch, and J. M. Pairot, "The ATV rendezvous predevelopment (ARP) project," ESA Directorate of Manned Spaceflight and Microgravity, Noordwijk, The Netherlands: European Space Agency Bulletin 89, February 1997.

[14] ESA Press Release (October 2005), "Safety and autonomy make the ATV unique," Available: www.esa.int/SPECIALS/ATV/SEMSEV5Y3EE_2.html.

[15] H. Schaub and J. L. Junkins, *Analytical Mechanics of Space Systems*, 1st Edition. Washington, D.C.: American Institute of Aeronautics and Astronautics, AIAA Education Series, 2003, pp. 593–599.

[16] NASA Johnson Space Center (October 2006), "ISS Payload Information Source: ISS Environment," website. Available: www.stationpayloads.jsc.nasa.gov/D-aboutiss/D6.html.

[17] J. Valasek, K. Gunnam, J. Kimmett, M. D. Tandale, J. L. Junkins, and D. Hughes, "Vision based sensor and navigation system for autonomous air refueling," *Journal of Guidance, Control, and Dynamics*, vol. 28, no. 5, pp. 979–989, September-October 2005.

[18] H. Schaub and J. L. Junkins, *Analytical Mechanics of Space Systems*, 1st Edition. Washington, D.C.: American Institute of Aeronautics and Astronautics, AIAA Education Series, 2003, pp. 127–136.

[19] The National Research Council: Committee on the Future of the Global Positioning System, *The Global Positioning System: A Shared National Asset*, Washington, D.C.: National Academies Press, 1995.

[20] S. I. Roumeliotis, A. E. Johnson, and J. F. Montgomery, "Augmenting inertial navigation with image-based motion estimation," in *Proc. 2002 IEEE International Conference on Robotics and Automation*, Washington, D.C., 11-15 May 2002, pp. 4326–4333.

[21] JenaOptronik GmbH (October 2006), "RVS datasheet," online. Available: www.jenaoptronik.de/ content/aocs/rvs/rvs.pdf.

[22] J. L. Junkins, H. Schaub, and D. Hughes, "Noncontact position and orientation measurement system and method," U.S. Patent 6,266,142, 2001.

[23] M. D. Tandale, R. Bowers, and J. Valasek, "Robust trajectory tracking controller for vision based autonomous aerial refueling of unmanned aircraft," *Journal of*

Guidance, Control, and Dynamics, vol. 29, no. 4, pp. 846–857, July-August 2006.

[24] J. Valasek, J. Kimmett, D. Hughes, K. Gunnam, and J. L. Junkins, "Vision based sensor and navigation system for autonomous aerial refueling," in *Proc. 1st AIAA Conference and Workshop on Unmanned Aerospace Vehicles, Systems, Technologies, and Operations*, Portsmouth, VA, 12-14 May 2002, AIAA paper 2002-3441.

[25] J. Kimmett, J. Valasek, and J. L. Junkins, "Autonomous aerial refueling utilizing a vision based navigation system," in *Proc. AIAA Guidance, Navigation, and Control Conference and Exhibit*, Monterey, CA, 5-8 August 2002, AIAA paper 2002-4469.

[26] J. Kimmett, J. Valasek, and J. L. Junkins, "Vision based controller for autonomous aerial refueling," in *Proc. IEEE Control Systems Society Conference on Control Applications*, Glasgow, Scotland, 18-20 September 2002, IEEE paper CCA02-CCAREG-1126.

[27] B. Iannotta, "Satellites that make service calls," *Aerospace America*, vol. 40, no. 8, pp. 36–39, August 2002.

[28] R. Alonso, J. L. Crassidis, and J. L. Junkins, "Vision-based relative navigation for formation flying of spacecraft," in *Proc. AIAA Guidance, Navigation, and Control Conference and Exhibit*, Denver, CO, 14-17 August 2000, AIAA paper 2000-4439.

[29] J. L. Junkins, D. Hughes, K. P. Wazni, and V. Pariyapong, "Vision-based navigation for rendezvous, docking, and proximity operations," in *Proc. 22nd AAS Guidance and Control Conference*, Breckenridge, CO, 3-7 February 1999, AAS paper 99-021.

[30] R. Alonso, J.-Y. Du, D. Hughes, J. L. Junkins, and J. L. Crassidis, "Relative navigation for formation flying of spacecraft," in *Proc. Flight Mechanics Symposium*, Greenbelt, MD, 19-21 June 2001.

[31] J.-Y. Du, "Visnav system description technical document," (unpublished; no date).

[32] H. Schaub and J. L. Junkins, *Analytical Mechanics of Space Systems*, 1st Edition. Washington, D.C.: American Institute of Aeronautics and Astronautics, AIAA Education Series, 2003.

[33] B. Wood, Y. Ding, and J. Valasek, "Simulator control via wireless data glove," in *Proc. AIAA Modeling and Simulation Technologies Conference*, Austin, TX, 11-14 August 2003, AIAA paper 2003-5523.

[34] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems*, Boca Raton, FL: Chapman & Hall/CRC, Applied Mathematics and Nonlinear Science Series, 2004, pp. 7–14; 24–34.

[35] D. C. Montgomery and G. C. Runger, *Applied Statistics and Probability for Engineers*, Third Edition. New York: John Wiley & Sons, Inc., 2003.

[36] J. L. Crassidis and J. L. Junkins, *Optimal Estimation of Dynamic Systems*, Boca Raton, FL: Chapman & Hall/CRC, Applied Mathematics and Nonlinear Science Series, 2004.

[37] F. L. Lewis and V. L. Syrmos, *Optimal Control*, Second Edition. New York: John Wiley & Sons, Inc., 1995.

[38] G. F. Franklin, J. D. Powell, and M. Workman, *Digital Control of Dynamic Systems*, Third Edition. Menlo Park, CA: Addison Wesley Longman, Inc., 1998.

[39] J. Valasek, "Digital control of aerospace systems," course notes, Department of Aerospace Engineering, Texas A&M University, Fall 2003 (unpublished).

[40] J. R. Broussard, "Design, implementation, and flight testing of PIF autopilots for general aviation aircraft," NASA Contractor Report 3709, July 1983.

[41] R. C. Nelson, *Flight Stability and Automatic Control*, Second Edition. St. Louis, MO: McGraw-Hill Higher Education, 1998.

[42] R. H. Vassar and R. B. Sherwood, "Formationkeeping for a pair of satellites in a circular orbit," *Journal of Guidance, Control, and Dynamics*, vol. 8, no. 2, pp. 235–242, March–April 1985.

[43] R. H. Battin, *An Introduction to the Mathematics and Methods of Astrodynamics*, Revised Edition. Washington, D.C.: American Institute of Aeronautics and Astronautics, AIAA Education Series, 1999.

[44] Honeywell Defense and Space Systems (May 2008), "Constellation series reaction wheels datasheet," online. Available: www51.honeywell.com/aero/common/documents/Constellation_Series_Reaction_Wheels.pdf.

[45] W. S. Levine, *Control System Applications*, Boca Raton, FL: CRC Press, 2000, pp. 132–133.

VITA

Jeffery Christopher Morris graduated from Geneva High School in Geneva, Alabama in 1999. His undergraduate studies were done at The University of Alabama in Tuscaloosa, Alabama, from which he received his Bachelor of Science in Aerospace Engineering in 2003. Jeffery then enrolled at Texas A&M University, completing all of his Masters coursework in 2006. After accepting employment at Odyssey Space Research, LLC, in Houston, Texas in 2006, where he is still employed, he continued to work on his research and thesis on a part-time basis. He completed all remaining requirements and obtained his Master of Science in Aerospace Engineering from A&M in 2009. He may be contacted at the following address:

1120 NASA Parkway, Suite 505

Houston, TX 77058